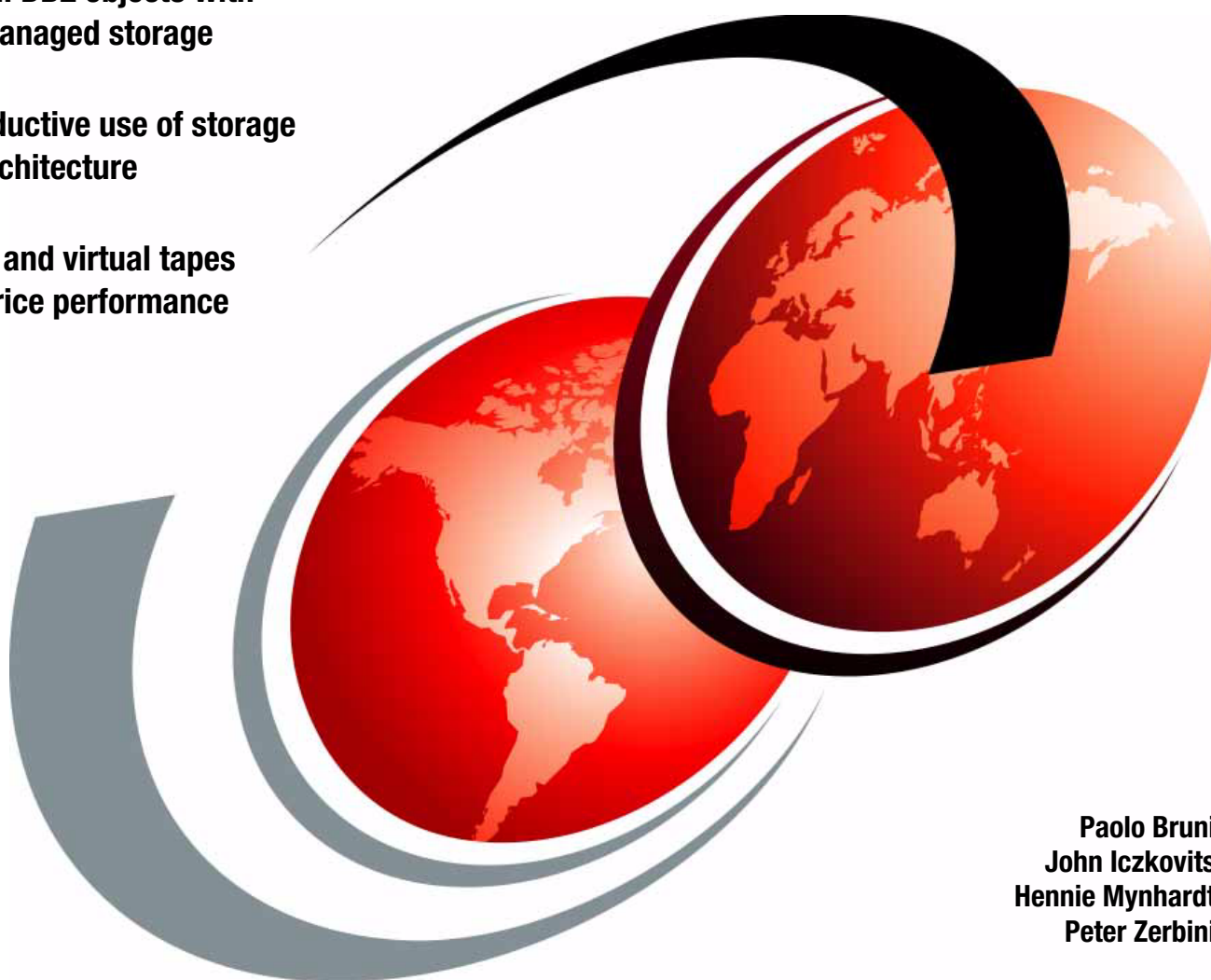


# DB2 9 for z/OS and Storage Management

Manage all DB2 objects with system-managed storage

Make productive use of storage servers architecture

Use tapes and virtual tapes for best price performance



Paolo Bruni  
John Iczkovits  
Hennie Mynhardt  
Peter Zerbini

**Redbooks**





International Technical Support Organization

**DB2 9 for z/OS and Storage Management**

September 2010

**Note:** Before using this information and the product it supports, read the information in “Notices” on page xvii.

**First Edition (September 2010)**

This edition applies to Version 9.1 of IBM DB2 for z/OS (program number 5635-DB2).

**© Copyright International Business Machines Corporation 2010. All rights reserved.**

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



# Contents

<b>Figures</b> .....	ix
<b>Tables</b> .....	xiii
<b>Examples</b> .....	xv
<b>Notices</b> .....	xvii
Trademarks .....	xviii
<b>Preface</b> .....	xix
The team who wrote this book .....	xix
Now you can become a published author, too! .....	xxii
Comments welcome. ....	xxii
Stay connected to IBM Redbooks .....	xxiii
<b>Chapter 1. Introduction</b> .....	1
1.1 Background .....	2
1.2 Summary of considerations .....	3
1.2.1 DB2 and storage management .....	3
1.2.2 DB2 and storage servers .....	6
<b>Chapter 2. Disk and tape environment overview</b> .....	11
2.1 How the new storage server technologies interact .....	12
2.2 Evolution of storage subsystems architecture .....	15
2.3 Disk capacity .....	15
2.3.1 Large volumes before z/OS V1R10 .....	16
2.3.2 Extended address volume (EAV) .....	18
2.3.3 Data sets eligible for EAV volumes .....	20
2.3.4 EAV volumes and multicylinder units .....	21
2.4 zArchitecture data scalability .....	22
2.4.1 Multiple allegiance .....	24
2.4.2 Parallel access volume (PAV) .....	25
2.4.3 z/OS Workload Manager: dynamic PAV tuning .....	27
2.4.4 HyperPAV .....	29
2.4.5 I/O priority queuing .....	32
2.4.6 Dynamic volume expansion .....	33
2.4.7 Storage management hardware .....	34
2.5 Redundant Array of Independent Disks (RAID) .....	35
2.5.1 RAID 6 overview .....	37
2.5.2 RAID 10 overview .....	38
2.5.3 RAID implementation in the DS8000 .....	39
2.6 HyperSwap .....	41
2.7 Seascape architecture .....	43
2.8 Architecture of the DS8000 series .....	44
2.9 Disk virtualization .....	46
2.9.1 Storage system virtualization .....	46
2.9.2 Abstraction layers for disk virtualization .....	47
2.9.3 Array sites .....	48
2.9.4 Arrays .....	49

2.9.5 Ranks	50
2.9.6 Extent pools	51
2.9.7 Logical volumes	53
2.9.8 Allocation, deletion, and modification of CKD volumes	55
2.9.9 Data striping by access method	57
2.9.10 Dynamic volume expansion	59
2.9.11 Logical subsystems	60
2.9.12 Summary of the virtualization hierarchy	62
2.9.13 Benefits of virtualization	63
2.10 Caching algorithms in DS8000	64
2.10.1 Sequential adaptive replacement cache (SARC)	64
2.10.2 Adaptive Multi-stream Prefetching (AMP)	66
2.10.3 Intelligent Write Caching	67
2.11 Common performance considerations for System z	68
2.11.1 Host connections to System z servers	68
2.11.2 DS8000 size compared to older storage subsystems	73
2.11.3 DS8000 processor memory size	73
2.11.4 Channels	74
2.11.5 Ranks and extent pool configuration	74
2.12 Copy Services	78
2.12.1 Copy Services with ESS	78
2.12.2 Copy Services with DS8000	82
2.13 Solid-state drives with DS8000	102
2.13.1 Solid-state drives overview	102
2.13.2 Easy Tier	103
2.14 Tapes: Storage Tier 3 in z/OS environment	104
2.14.1 Tape virtualization	106
2.14.2 TS7700 Virtualization Engine	109
2.15 Partitioned data sets	111
2.15.1 PDS	111
2.15.2 PDSE	112
2.16 New technologies and functions benefit DB2 for z/OS	114
<b>Chapter 3. DB2 storage objects</b>	<b>115</b>
3.1 DB2 overview	116
3.2 Non-VSAM data sets	116
3.3 DB2 data objects	117
3.3.1 Table	117
3.3.2 Table space	118
3.3.3 Index	124
3.3.4 Index space	125
3.3.5 Database	125
3.3.6 Storage group	125
3.4 Creating table spaces and index spaces	127
3.4.1 DB2-defined and managed	127
3.4.2 User defined and managed	128
3.4.3 DB2 table spaces on EAV	128
3.4.4 VSAM striping guidelines for DB2 table spaces	130
3.5 Large sequential data sets	132
3.6 DB2 system table spaces	136
3.6.1 The DB2 catalog and directory	136
3.6.2 The WORKFILE database	137
3.6.3 SYSIBM.SYSCOPY	142

3.6.4	SYSIBM.SYSLGRNX .....	144
3.7	DB2 application table spaces .....	145
3.8	Index compression .....	145
3.9	DB2 recovery data sets .....	147
3.9.1	Bootstrap data sets .....	147
3.9.2	Active logs .....	148
3.9.3	Archive logs .....	150
3.9.4	Image copies .....	152
3.9.5	TOKEN .....	161
3.9.6	DFSMS COPYPOOL .....	162
3.9.7	DUMPCLASS .....	163
3.9.8	Object-level backups .....	163
3.10	DFSMSShsm and DB2 BACKUP SYSTEM .....	165
3.10.1	BACKUP SYSTEM utility .....	165
3.10.2	RESTORE SYSTEM utility .....	168
3.11	Online CHECK DATA .....	170
3.12	Other DB2 data sets .....	171
3.12.1	DB2 library data sets .....	172
3.12.2	DB2 temporary and SORT data sets .....	172
3.13	DB2 data sets naming conventions .....	173
3.13.1	Table space and index space names .....	173
3.13.2	BSDS names .....	174
3.13.3	Active log names .....	174
3.13.4	Archive log and BSDS backup names .....	175
3.13.5	Image copy names .....	175
3.13.6	Determining which DB2 data sets reside on the same disk .....	175
3.13.7	DSNZPARMs affecting DB2 data set sizes .....	176
3.13.8	Utility work data sets .....	187
3.14	Format and preformat of DB2 data sets .....	188
<b>Chapter 4.</b>	<b>System managed DB2 data .....</b>	<b>193</b>
4.1	Concepts and components .....	194
4.1.1	Evolution .....	194
4.1.2	DFSMS/MVS components .....	194
4.1.3	Benefits .....	202
4.2	Storage management with DFSMS .....	203
4.2.1	Introduction .....	203
4.2.2	Automatic class selection (ACS) routines .....	205
4.2.3	SMS classes .....	207
4.2.4	Naming standards .....	217
4.3	SMS examples for DB2 databases .....	218
4.3.1	Using ISMF to display SMS constructs .....	218
4.3.2	SMS data class .....	219
4.3.3	SMS storage class .....	222
4.3.4	SMS management class .....	229
4.3.5	SMS storage groups .....	230
4.3.6	DB2 STOGROUPs and SMS storage groups .....	232
4.3.7	Assigning SMS classes and storage groups for DB2 objects .....	233
4.3.8	SMS base configuration .....	235
4.3.9	Separation profile .....	235
<b>Chapter 5.</b>	<b>Implementing SMS for a DB2 environment .....</b>	<b>237</b>
5.1	Considerations before implementing SMS .....	238

5.1.1	DB2 data set considerations . . . . .	238
5.1.2	Environment considerations . . . . .	238
5.2	Implementing SMS constructs . . . . .	239
5.2.1	Data Class implementation . . . . .	239
5.2.2	Storage Class implementation . . . . .	258
5.2.3	Management Class implementation . . . . .	269
5.2.4	Storage group implementation . . . . .	273
5.2.5	Separation Profile . . . . .	289
5.3	Considerations for DB2 data set types . . . . .	293
5.3.1	Active logs and BSDS . . . . .	293
5.3.2	DB2 catalog and directory . . . . .	294
5.3.3	DB2 user table spaces and indexes . . . . .	294
5.3.4	Archive log data sets on disk . . . . .	295
5.3.5	Image copy data sets on disk . . . . .	296
5.3.6	DB2 work file data sets (DSNDB07 or equivalent) . . . . .	297
5.4	Converting from non-SMS to SMS . . . . .	298
5.4.1	CONVERTV approach . . . . .	302
5.4.2	COPY approach . . . . .	308
5.4.3	Overview of the DB2 and SMS relationship . . . . .	310
5.4.4	Advantages of SMS managing DB2 data . . . . .	310
5.4.5	SMS management goals . . . . .	311
5.4.6	Positioning for implementation . . . . .	311
5.4.7	Conversion Process . . . . .	313
5.4.8	DFSMS FIT . . . . .	315
5.4.9	NaviQuest . . . . .	316
<b>Chapter 6.</b>	<b>DB2 I/O performance and monitoring . . . . .</b>	<b>317</b>
6.1	Global view of a DB2 I/O . . . . .	318
6.2	Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS . . . . .	320
6.2.1	Accounting I/O information . . . . .	321
6.2.2	Statistics I/O information . . . . .	322
6.2.3	Performance I/O information and I/O activity . . . . .	324
6.3	RMF monitoring . . . . .	325
6.3.1	I/O response time . . . . .	326
6.3.2	PAV . . . . .	327
6.3.3	IOP/SAP . . . . .	327
6.3.4	FICON host channel . . . . .	328
6.3.5	FICON director . . . . .	329
6.3.6	Cache and NVS . . . . .	329
6.3.7	FICON/Fibre port and host adapter . . . . .	332
6.3.8	Extent pool and rank . . . . .	334
6.4	RMF Magic for Windows . . . . .	336
<b>Chapter 7.</b>	<b>DB2 I/O operations . . . . .</b>	<b>339</b>
7.1	Overview of I/O options . . . . .	340
7.1.1	Avoiding I/O operations . . . . .	340
7.1.2	How I/O is performed in DB2 . . . . .	340
7.2	Data read operations . . . . .	342
7.2.1	Normal read . . . . .	342
7.2.2	Sequential prefetch . . . . .	343
7.2.3	Dynamic prefetch . . . . .	344
7.2.4	List prefetch . . . . .	345
7.2.5	Prefetch quantity . . . . .	346

7.2.6 Data manager threshold . . . . .	348
7.2.7 Sequential prefetch threshold . . . . .	348
7.3 Data write operations . . . . .	349
7.3.1 Asynchronous writes . . . . .	351
7.3.2 Synchronous writes . . . . .	351
7.3.3 Immediate write threshold (IWTH) . . . . .	352
7.3.4 Deferred write threshold (DWQT) . . . . .	352
7.3.5 Write quantity . . . . .	353
7.3.6 Tuning buffer pool write frequency . . . . .	353
7.4 Log writes . . . . .	358
7.4.1 Log sizing parameters . . . . .	358
7.4.2 Improving log write performance . . . . .	359
7.4.3 Asynchronous writes . . . . .	361
7.4.4 Synchronous writes . . . . .	361
7.4.5 Writing to two logs . . . . .	361
7.5 Log reads . . . . .	362
7.5.1 Improving log-read performance . . . . .	362
7.5.2 Active log size . . . . .	363
7.6 Distributing data sets more efficiently . . . . .	365
7.6.1 Creating additional work file table spaces to reduce contention . . . . .	366
7.6.2 Formatting early and speeding-up formatting . . . . .	367
7.7 DB2 sort work files . . . . .	369
7.8 DB2 and tape processing . . . . .	370
7.8.1 Virtual tape terminology . . . . .	370
7.8.2 Virtual volume size versus physical volume size . . . . .	371
7.8.3 DSNZPARMs for archiving to tape . . . . .	372
7.8.4 Large Block Interface . . . . .	374
7.8.5 Conclusion . . . . .	374
<b>Appendix A. Frequently asked questions . . . . .</b>	<b>375</b>
A.1 What are the components of I/O response time? . . . . .	376
A.2 I hear that DB2 does not work with SMS; is that true? . . . . .	378
A.3 Is it true that only my DB2 user data, not system data, can be SMS-managed? . . . . .	378
A.4 If my volumes are SMS-managed, should I worry about space? . . . . .	378
A.5 How do I determine names and available space for SMS-managed volumes? . . . . .	378
A.6 How many storage groups should I have in my production environment? . . . . .	379
A.7 How many storage groups should I have in non-production? . . . . .	380
A.8 What are considerations for consolidating to large volumes: Mod 27 or 54 . . . . .	381
A.9 How did these space allocations happen? . . . . .	381
A.10 My storage administrator sees many disk-write bursts. How can I help? . . . . .	381
A.11 Do I need PDSEs? . . . . .	382
A.12 Why do I have space anomalies? . . . . .	382
A.13 What about catalogs? . . . . .	384
A.14 Do I need to let my storage administrator know anything about DB2 V8? . . . . .	384
A.15 What is extent reduction? . . . . .	384
A.16 How much data can I really lose? . . . . .	385
A.17 What about pseudo deletes? . . . . .	385
A.18 What are FlashCopy and SnapShot? . . . . .	386
A.19 How many buffer pools do I need? . . . . .	386
A.20 What guidelines exist for sizing structures in the CF regarding the GBP? . . . . .	387
A.21 What is a good way to establish a buffer pool strategy? . . . . .	387
<b>Abbreviations and acronyms . . . . .</b>	<b>389</b>

<b>Related publications</b> .....	393
IBM Redbooks .....	393
Other publications .....	394
Online resources .....	394
How to get Redbooks .....	395
Help from IBM .....	395
<b>Index</b> .....	397

# Figures

1-1 DS8000 and Seascope architecture . . . . .	7
1-2 IBM System Storage Virtualization Engine TS7700 solution . . . . .	9
2-1 Traditional DASD capacity . . . . .	16
2-2 Large volume support . . . . .	17
2-3 EAV Volumes . . . . .	19
2-4 EAV and multicylinder units . . . . .	21
2-5 zArchitecture data scalability . . . . .	23
2-6 Parallel I/O capability with multiple allegiance . . . . .	24
2-7 Traditional z/OS behavior . . . . .	26
2-8 z/OS behavior with PAV . . . . .	27
2-9 WLM assignment of alias addresses . . . . .	28
2-10 Dynamic PAVs in a sysplex . . . . .	28
2-11 Parallel access volumes: basic operational characteristics . . . . .	30
2-12 HyperPAV: basic operational characteristics . . . . .	31
2-13 I/O priority queuing . . . . .	33
2-14 RAID . . . . .	37
2-15 One RAID 6 stride . . . . .	38
2-16 Basic HyperSwap . . . . .	42
2-17 DS8000 architecture . . . . .	45
2-18 Storage facility virtualization . . . . .	47
2-19 Physical layer as the base for virtualization . . . . .	48
2-20 Array sites . . . . .	49
2-21 Creation of a array . . . . .	50
2-22 Forming an FB rank with 1 GB extents . . . . .	51
2-23 Extent pools . . . . .	53
2-24 Allocation of a CKD logical volume . . . . .	54
2-25 Extent allocation methods . . . . .	56
2-26 Free extents can be used for standard and extent space-efficient volumes . . . . .	57
2-27 Grouping of volumes in LSSs . . . . .	61
2-28 Logical storage subsystem . . . . .	62
2-29 Virtualization hierarchy . . . . .	63
2-30 Sequential Adaptive Replacement Cache . . . . .	65
2-31 Intelligent Write Caching . . . . .	68
2-32 DS8100 front-end connectivity example, partial view . . . . .	69
2-33 zHPF Link Protocol Comparison for a 4KB READ . . . . .	70
2-34 Channel command word . . . . .	71
2-35 Transferring an 8 KB record from and to an EF data set . . . . .	72
2-36 Extent pool affinity to processor complex with one extent pool for each rank . . . . .	75
2-37 Extent pool affinity to processor complex with pooled ranks in two extent pools . . . . .	76
2-38 Mix of extent pools . . . . .	77
2-39 ESS Copy Services . . . . .	78
2-40 DS8000 Copy Services Matrix . . . . .	83
2-41 FlashCopy concept . . . . .	85
2-42 Incremental FlashCopy . . . . .	88
2-43 Data set FlashCopy . . . . .	89
2-44 Multiple Relationship FlashCopy . . . . .	89
2-45 Consistency Group FlashCopy . . . . .	90
2-46 Establish FlashCopy on existing Metro Mirror (MM) or Global Copy (GC) primary . . . . .	91

2-47 Remote Pair FlashCopy . . . . .	92
2-48 Metro Mirror basic operation . . . . .	94
2-49 Global Copy basic operation . . . . .	95
2-50 Global Mirror basic operation . . . . .	96
2-51 How Global Mirror works . . . . .	97
2-52 Metro/Global Mirror elements . . . . .	98
2-53 Metro/Global Mirror overview diagram . . . . .	99
2-54 z/OS Metro Global Mirror . . . . .	100
2-55 IBM Tape solutions . . . . .	105
2-56 Main components of a tape virtualization solution . . . . .	107
3-1 CREATE STOGROUP syntax diagram with SMS attributes . . . . .	125
3-2 Empty volume . . . . .	129
3-3 Full volume after inserts . . . . .	130
3-4 New RMF format for EAV devices . . . . .	130
3-5 Installation panel DSNTIPD . . . . .	137
3-6 Install panel for WORK FILE DATABASE . . . . .	138
3-7 Data parameters for DB2 installation panel 1 . . . . .	141
3-8 Installation panel DSNTIPA3 for defining subsystem databases and data sets . . . . .	142
3-9 DB2 log and its data sets . . . . .	149
3-10 DSNTIP6 installation panel: specifying system-level backups . . . . .	164
3-11 Database and storage integration . . . . .	165
3-12 DB2 Check data using FlashCopy V2 . . . . .	171
3-13 Specifying SORT options during installation - Panel DSNTIP6 . . . . .	172
3-14 Sliding scale for 64 GB data sets . . . . .	178
3-15 Sliding scale for 16 GB data sets . . . . .	179
3-16 Allocations with no system settings . . . . .	181
3-17 Allocations with system settings . . . . .	183
3-18 Preformat example . . . . .	190
4-1 ISMF Primary option menu for users . . . . .	195
4-2 ISMF Primary option menu for storage administrator . . . . .	196
4-3 DFSMSHsm components . . . . .	199
4-4 Implementing an SMS configuration . . . . .	205
4-5 ACS routine execution process . . . . .	206
4-6 SMS Construct Relationship . . . . .	207
4-7 Storage administrator panel . . . . .	219
4-8 Data CClass Display, panel five of five . . . . .	220
4-9 Data Class DB2EFEA - page 2 of 5. All other settings were default . . . . .	225
4-10 Storage Class DB9ASDR . . . . .	226
4-11 ACS test output example . . . . .	234
4-12 ACS routine extract using table and index name filter list . . . . .	234
5-1 ISMF panel for Data Class . . . . .	240
5-2 Data Class page 1 of 5: Alter DB2EFEAX . . . . .	240
5-3 Data Class page 2 of 5: Alter DB2EFEAX . . . . .	241
5-4 Data Class page 3 of 5: Alter DB2EFEAX . . . . .	242
5-5 Data Class page 4 of 5: Alter DB2EFEAX . . . . .	242
5-6 Data Class page 5 of 5: Alter DB2EFEAX . . . . .	243
5-7 Data Class page 1 of 5: Alter PE10PO . . . . .	244
5-8 Data Class page 2 of 5: Alter PE10PO . . . . .	245
5-9 Data Class PE10VS: Page 1 of 5 . . . . .	247
5-10 Data Class PE10VS: Page 2 of 5 . . . . .	247
5-11 Data Class PE10VS: Page 3 of 5 . . . . .	248
5-12 Data Class PE10VS - page 4 of 5 . . . . .	248
5-13 Data Class PE10VS: Page 5 of 5 . . . . .	249



5-14 Data Class DB2EFEAS: Page 1 of 5	253
5-15 Data Class DB2EFEAS: Page 2 of 5	253
5-16 LISTCAT after REPRO	255
5-17 LISTCAT after more REPROs	255
5-18 LISTCAT after more REPROs	256
5-19 LISTCAT after more REPROs	256
5-20 Data Class WELCGLRG	257
5-21 Initial Storage Class panel	259
5-22 Storage Class: Page 1	260
5-23 Storage Class: Page 2 of 2	260
5-24 Alter of Data Class DB9AARCH: Enable EF and compaction	265
5-25 Data Class DB9AARCH is altered to disable compaction, but striping the data set	267
5-26 Initial Management Class page 1	270
5-27 Alter the Management Class DB9AML1	270
5-28 Alter the storage group DB9AML1	271
5-29 ISPF 3.4 of the migrated image copy data set	273
5-30 Storage group: initial panel	274
5-31 Storage Group Alter: page 1 of 2	275
5-32 Storage Group Alter: page 2 of 2	275
5-33 Panel to specify SMS Storage Group Status	276
5-34 Page 1 of 2 to add volumes	276
5-35 Page 2 of 2 to add volumes	277
5-36 Storage Group DB9ATST1: pages 1 and 2	278
5-37 Storage Group DB9ATST2: pages 1 and 2	279
5-38 LISTVOL output for storage group DB9ATST1	279
5-39 LISTVOL output for storage group DB9ATST2	280
5-40 Storage Group panels for DB9ATST3 and DB9AT3OV	282
5-41 LISTVOL for Storage Group DB9ATST3	283
5-42 LISTVOL for Storage Group DB9AT3OV	283
5-43 Storage Class DB9ATST4 with Multi-Tiered SGs (storage groups) enabled	285
5-44 Storage Group DB9AT4T1	286
5-45 Storage Group DB9AT4T2	287
5-46 ISMF LISTVOL display for JIAE12 (DB9AT4T1) and JIAF01 (DB9AT4T2)	288
5-47 ISPF 3.4 display of data sets and volumes for multitiered storage groups	289
5-48 Alter of the base configuration	290
5-49 SCDS base Alter panel	291
5-50 ISMF option 7 shows the ACS panel	299
5-51 IACS TEST SELECTION panel	300
5-52 Test the DSN option	300
5-53 Test ACS routines panel	301
5-54 ACS routine test results	301
5-55 ISMF option 7, then option 5: Display ACS Object Information	302
6-1 Scope of performance analysis tools	318
6-2 Installation panel DSNTIPN	320
6-3 OMEGAMON PE accounting, buffer pool section	321
6-4 OMEGAMON PE accounting class 3 times	322
6-5 OMEGAMON PE statistics, buffer pool read operations section	323
6-6 OMEGAMON PE statistics, log activity section	324
6-7 Buffer pool section from I/O activity summary report	325
6-8 I/O queuing activity report	327
6-9 Cache subsystem summary report	329
6-10 Cache subsystem overview report	330
6-11 Cache Subsystem Activity by volume serial number	331

6-12 DS8000 port numbering .....	333
7-1 Diagram of DB2 storage .....	342
7-2 DB2 sequential prefetch .....	343
7-3 Install Panel for log data sets .....	358
7-4 Log record path to disk .....	361
7-5 DB2 installation panel for ARCHIVE LOG data set parameters .....	372

# Tables

3-1	Maximum column and row length per page for tables . . . . .	117
3-2	Table space sizes and total space sizes. . . . .	120
3-3	DB2 STOGROUPs and SMS storage groups differences . . . . .	126
3-4	Acceptable values for the data values in Figure 3-8 . . . . .	142
3-5	SYSIBM.SYSCOPY: ICTYPE column values . . . . .	143
3-6	DB2 Image Copy with and without concurrent COPY . . . . .	157
3-7	DB2 token breakout and description . . . . .	161
3-8	Maximum allocation for sliding secondary extents - without volume fragmentation. .	183
3-9	Utilities that can benefit DFSMS striping. . . . .	188
4-1	Differences between DB2 STOGROUP and SMS storage group. . . . .	232
6-1	Trace requirement for the I/O activity reports . . . . .	325
7-1	The number of pages read by prefetch by buffer pool size . . . . .	347
7-2	Number of pages that DB2 can write in a single I/O operation . . . . .	349
7-3	Number of pages that DB2 can write to a single I/O operation for utility-writes . . . .	350
7-4	The number of pages that DB2 can write for a single I/O operation for LOB writes . .	350
7-5	Number of changes pages based on buffer pool size . . . . .	353
7-6	Number of pages that DB2 can write in a single I/O operation . . . . .	353



# Examples

3-1 DDL of with DSVCI enabled . . . . .	122
3-2 VSAM LISTCAT output of TS with DSVCI enabled . . . . .	123
3-3 DDL with DSVCI disabled . . . . .	123
3-4 IDCAMS LISTCAT output of TS with DSVCI disabled . . . . .	124
3-5 DDL for STOGROUP creation with DATA CLASS specified . . . . .	126
3-6 DB2 catalog information for a STOGROUP . . . . .	126
3-7 CREATE TABLESPACE DDL: DB2-defined . . . . .	127
3-8 IDCAMS LISTCAT output of a DB2 defined TS . . . . .	128
3-9 User-defined table space: Step 1, define the cluster . . . . .	128
3-10 User-defined table space: Step 2, define the table space . . . . .	128
3-11 Sequence for allocating table spaces in EAV . . . . .	129
3-12 COPY utility requesting 7,510 cylinders primary without LARGE specified . . . . .	133
3-13 COPY utility: TEMPLATE, no space requested, and LARGE not specified . . . . .	133
3-14 COPY utility requesting 7,510 cylinders primary with LARGE specified . . . . .	134
3-15 COPY utility: TEMPLATE, no space requested, Data Class with LARGE enabled . . . . .	135
3-16 Example of a COPY with TEMPLATE . . . . .	154
3-17 COPY utility: LIMIT keyword in TEMPLATE . . . . .	155
3-18 COPYTOCOPY example with TEMPLATE usage . . . . .	156
3-19 COPY using CLONE option . . . . .	156
3-20 CONCURRENT COPY output, with error . . . . .	158
3-21 BSDS BACKUP SYSTEM utility history output from DSNJU004 . . . . .	162
3-22 BACKUP SYSTEM JCL . . . . .	167
3-23 FAST REPLICATION output in DFSMSshm for the DB COPYPOOL . . . . .	167
3-24 FAST REPLICATION output in DFSMSshm for the LG COPYPOOL . . . . .	167
3-25 BSDS BACKUP SYSTEM UTILITY HISTORY output from DSNJU004 . . . . .	168
3-26 Create a system restart record using DSNJU003 JCL . . . . .	168
3-27 Output of DSNJU003 to create a SYSPITR record in the BSDS . . . . .	169
3-28 RESTORE SYSTEM sample JCL . . . . .	169
3-29 RESTORE SYSTEM job output . . . . .	169
3-30 Recovery messages in DFSMSshm log . . . . .	169
3-31 LISTCAT output for checking allocations . . . . .	184
3-32 Extend trace output . . . . .	185
3-33 Extended format table space with no Guaranteed Space assigned after CREATE . . . . .	191
3-34 Successful DFSMSdss RELEASE job after the table space was stopped . . . . .	192
3-35 LISTCAT output after RELEASE was successfully executed . . . . .	192
4-1 DFSMSdss job that deletes archive logs after 15 days, not referenced . . . . .	197
4-2 Create a storage group syntax . . . . .	221
4-3 Original allocation non-striped . . . . .	223
4-4 LISTCAT after the data set is striped . . . . .	224
4-5 Image copy with a non-striped sequential data set . . . . .	227
4-6 Image copy with a non-striped sequential data set . . . . .	227
4-7 Image copy with a striped sequential data set . . . . .	228
5-1 LISTCAT for Data Class DB2EFEAX . . . . .	243
5-2 IEFBR14 example using Data Class . . . . .	245
5-3 Example of overriding the LRECL and trying to override the block size . . . . .	246
5-4 DEFINE using Data Class PE10VS . . . . .	249
5-5 LISTCAT output after using Data Class PE10VS . . . . .	250
5-6 DEFINE using Data Class PE10VS with an override of the space parameter . . . . .	251

5-7	LISTCAT output after using Data Class PE10VS with an override of space . . . . .	251
5-8	Example of DEFINE using Data Class DB2EFEAS . . . . .	254
5-9	LISTCAT of data set DB9AU.DSNDBD.TEST.VOLCOUNT . . . . .	254
5-10	Final REPRO . . . . .	257
5-11	Storage Class ACS routine specifying some non-SMS-managed data sets. . . . .	258
5-12	Striping active log data sets . . . . .	262
5-13	LISTCAT output for DB9AU.LOGCOPY1.DS04 . . . . .	263
5-14	DSNZPARMs used for our test of the archive log data sets: allocation . . . . .	265
5-15	Data and Storage Class examples for the archive log data sets . . . . .	266
5-16	LISTCAT and ISPF 3.4 of archive log data set striped and compacted . . . . .	266
5-17	LISCAT and ISPF 3.4 of archive log data set striped, with compaction disabled . . . . .	268
5-18	ACS routines for migration of image copy data sets. . . . .	271
5-19	Image copy migration . . . . .	272
5-20	Storage Class and Storage Group ACS routines for TEST1 and TEST2 data sets . . . . .	280
5-21	TEST1 data sets . . . . .	281
5-22	Storage Class and storage group ACS routines for TEST3 data sets . . . . .	283
5-23	ISPF 3.4 of the four data sets created . . . . .	284
5-24	ISMF volume panel for JIAC01 . . . . .	284
5-25	Storage Class and Group ACS routines for multitiered storage groups . . . . .	288
5-26	Separation Profile data set entry. . . . .	291
5-27	IDCAMS DEFINE of the DS05 log data sets . . . . .	292
5-28	DEFINE using Guaranteed Space, point to the same volume . . . . .	292
5-29	LISTCAT and PRINT of SMS-managed DB2 LDS . . . . .	298
5-30	Storage Class member SCACTIVE . . . . .	302
5-31	IEHLIST for volumes JI8118 . . . . .	303
5-32	CREATE STOGROUP DDL for non-SMS-managed volumes JI8118 and JI8119 . . . . .	303
5-33	CREATE TABLESPACE to fill up both volumes, except for 10 cylinders on each . . . . .	303
5-34	IEHLIST for non-SMS volume JI8118 after allocations. . . . .	304
5-35	IEHLIST for non-SMS volume JI8119 after allocations. . . . .	304
5-36	LISTCAT of one of the eight data sets created, no SMS class information . . . . .	305
5-37	CONVERTV with TEST option . . . . .	305
5-38	CONVERTV successfully completed . . . . .	306
5-39	EHLIST for volume JI8118 after CONVERTV. . . . .	307
5-40	IEHLIST for volume JI8119 after CONVERTV . . . . .	307
5-41	LISTCAT example of converted data set showing SMS classes . . . . .	308
5-42	COPY from non-SMS volume JI8118 and JI8119 to SMS-managed volumes . . . . .	308
6-1	RMF direct access device activity report. . . . .	326
6-2	Channel Path Activity report . . . . .	328
6-3	FICON Director Activity report . . . . .	329
6-4	Cache Device Activity report detail by volume . . . . .	332
6-5	DS8000 link statistics . . . . .	332
6-6	Rank statistics for one rank per extent pool . . . . .	334
6-7	Rank statistics for multiple ranks on one extent pool without SPS . . . . .	335
6-8	Rank statistics for multiple ranks on one extent pool using SPS . . . . .	336
7-1	DIS BUFFERPOOL with DETAIL output. . . . .	354
7-2	OMEGAMON PE buffer pool statistics trace report . . . . .	355
7-3	OMEGAMON PE database buffer pool statistics . . . . .	356
7-4	OMEGAMON statistics report: log activity . . . . .	364
A-1	ISMF Storage Group LISTVOL report . . . . .	378
A-2	ISPF 3.4 volume space listing . . . . .	379
A-3	MVS command to display storage group volumes . . . . .	379
A-4	Report of space anomalies: Question 1 . . . . .	383

# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

# Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	Magstar®	RMF™
AS/400®	MVS™	S/390®
CICS®	NetView®	System p®
DB2®	OMEGAMON®	System Storage®
DS6000™	OS/390®	System z10®
DS8000®	OS/400®	System z9®
ECKD™	Parallel Sysplex®	System z®
Enterprise Storage Server®	POWER5™	Tivoli®
ESCON®	POWER5+™	TotalStorage®
FICON®	pSeries®	z/Architecture®
FlashCopy®	RACF®	z/OS®
GDPS®	Redbooks®	z/VM®
HACMP™	Redpaper™	z10™
HyperSwap®	Redpapers™	z9®
IBM®	Redbooks (logo)  ®	zSeries®
IMS™	Resource Measurement Facility™	
iSeries®	RETAIN®	

The following terms are trademarks of other companies:

Snapshot, and the NetApp logo are trademarks or registered trademarks of NetApp, Inc. in the U.S. and other countries.

ACS, and the Shadowman logo are trademarks or registered trademarks of Red Hat, Inc. in the U.S. and other countries.

Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.



# Preface

This IBM® Redbooks® publication can help you tailor and configure DFSMS constructs to be used in an IBM DB2® 9 for z/OS® environment. In addition, it provides a broad understanding of new disk architectures and their impact in DB2 data set management for large installations.

This book addresses both the DB2 administrator and the storage administrator. The DB2 administrator can find information about how to use DFSMS for managing DB2 data sets; the storage administrator can find information about the characteristics of DB2 data sets and how DB2 uses the disks. This book describes optimal use of disk storage functions in DB2 for z/OS environments that can best make productive use of the synergy with I/O subsystem on IBM System z®.

This book covers the following topics:

- ▶ Using SMS to manage DB2 catalog, log, data, indexes, image copies, archives, work files
- ▶ Taking advantage of IBM FlashCopy® for DB2 utilities, striping, copy pools
- ▶ Setting page sizes and using sliding allocation
- ▶ A description of PAV, MA, MIDAW, EF, EA, EAV, zHPF and why they are helpful
- ▶ Compressing data and the use disk and tape for large data sets
- ▶ Backup and restore, and remote copy services

## The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

**Paolo Bruni** is a DB2 Information Management Project Leader at the International Technical Support Organization based in Silicon Valley Lab, San Jose, California. In this capacity, he has authored several IBM Redbooks publications on DB2 for z/OS and Data Management tools, and has conducted workshops and seminars worldwide. During Paolo's many years with IBM, in development and in the field, his work has been mostly related to database systems.

**John Iczkovits** is a Consulting IT Specialist with IBM Advanced Technical Skills. He provides DB2 for z/OS technical support and consulting services. His areas of expertise include DB2 data sharing, performance, and availability. His work at IBM includes experience supporting DB2 as a Systems Engineer, Database Administrator, IT Specialist, Consultant, and Project Leader. He has an operations background and more than 28 years of IT experience, ranging from database products such as DB2 and IMS™ to MVS™ systems support, including storage management. He has also co-authored IBM Redbooks, IBM Redpapers™ publications, and white papers, and he has presented at SHARE, the DB2 Technical Conference, the Information On Demand Conference, zEXPO Conference, and at local DB2 users groups.

**Hennie Mynhardt** is an IBM Senior Certified Consulting IT Specialist with the IBM Software Group. He has lead and worked on various technical projects for database customers in the U.S.A. and other countries. His special interests are systems performance tuning and backup/recovery. He currently provides technical consulting and pre- and post-sales support for DB2 for z/OS Engine and Tools. Hennie has co-authored *Securing and Auditing Data on DB2 for z/OS*, SG24-7720, *Optimizing Restore and Recovery Solutions with DB2 Recovery*

*Expert for z/OS V2.1, SG24-7606, and DB2 Recovery Expert for z/OS User Scenarios, SG24-7226.*

**Peter Zerbini** is a Certified Senior Software IT Architect with IBM Software Group in Germany. He has more than 36 years of experience in the IT area. He has a mainframe hardware background and more than 25 years in software. His areas of expertise include z/OS, storage, and information management software such as DB2. He has written extensively about storage domain and has co-authored several IBM Redbooks.



*From left to right: Hennie, Peter, Paolo, and John*

Thanks to the following people for their contributions to this project:

Rich Conway  
Bob Haimowitz  
Emma Jacobs  
Diane Sherman  
International Technical Support Organization

Charlie Burger  
IBM Sales & Distribution, ATS Americas

Jeff Sullivan  
IBM STG Systems Software

Babette Haeusser  
IBM ESCC, Germany

Victor Liang  
IBM STG, SMS Development

Wayne Rhoten  
IBM STG, DFSMS development

Rick Butler  
BMO Financial Group, Toronto

Jeff Berger  
Gopal Krishnan  
Dave Levis  
Bruce McAlister  
Roger Miller  
IBM Silicon Valley Lab

Thanks to the following authors of the previous related book, *Storage Management with DB2 for OS/390*, SG24-5462, published in September, 1999:

Paolo Bruni  
Hans Duerr  
Daniel Leplaideur  
Steve Wintle

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author - all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:  
[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. 1WLB Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on Facebook:  
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on twitter:  
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:  
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:  
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:  
<http://www.redbooks.ibm.com/rss.html>





# Introduction

This chapter provides a background of storage management and a summary of auxiliary storage considerations.

This chapter contains the following topics:

- ▶ Background
- ▶ Summary of considerations

## 1.1 Background

The auxiliary storage management in the DB2 environment for the z/OS platform has been mainly the responsibility of the database administrators.

In the first few years of its usage, DB2's implicit definition of page sets through its storage groups (STOGROUP) often replaced the more traditional method of explicitly allocating Virtual Storage Access Method (VSAM) data sets because of DB2's simplicity and ease of use.

Database administrators worried about separation of critical data sets, such as data from indexes, data from log, copies of log and bootstrap data set (BSDS), spreading work files, through the use of multiple storage groups and the careful association of volumes to storage groups. Operators, storage managers, system programmers and performance analysts had to interact frequently with the database administrators to resolve issues related to DB2 data set management. Furthermore, database administrators did not look favorably at System Managed Storage (SMS) space management because they believed that it interfered with the hand-placement of critical DB2 data sets; SMS usage was limited to certain hierarchical management of backup data sets (image copies and archived logs).

Today, a growing number of data warehousing applications are available that require very large table spaces and query parallelism. These types of applications create an expansion in the number of DB2 objects. Conversely, more flexible functions in SMS-related products have provided innovative methods for space and backup management. These functions can help most medium-to-large DB2 installations, typical of enterprise systems or warehouse applications that do not have the time to devote the considerable amount of resources necessary to manually manage several thousand DB2 objects.

Furthermore, as processors and disk control units have provided more capacity and more memory, DB2 exploits its larger buffer pools as a second level of cache for I/O execution, reducing the I/O frequency and making it mostly asynchronous. This approach implies that the criticality of data set placement is greatly reduced.

In this book, as a level set, first we examine DB2 data set and I/O characteristics, then we look at the main concepts and functions of SMS, and then at the recent evolution of storage servers (disks).

We provide a mapping of the possible applicability of SMS for all except the most critical applications. This approach allows the database administrators to concentrate on DB2 data sets that are relative to the applications with the highest service level requirements; the storage administrators can use SMS to simplify disk use and control.

We finally look at the impact that large cache and the virtual architecture of the current disk technology have on dealing with DB2 data.

Because of the necessity to monitor performance to avoid surprises, we also show how to look at DB2 and I/O performance tools output from the overall storage management perspective.



## 1.2 Summary of considerations

This book describes how to make productive use of auxiliary storage by DB2 for z/OS (DB2), analyzing two major areas:

- ▶ DB2 and storage management
- ▶ DB2 and storage servers

This section summarizes the major conclusions from our tests.

### 1.2.1 DB2 and storage management

A detailed analysis of the various types of DB2 data sets shows that DFSMS can automatically manage all of the data sets that DB2 uses and requires. However, certain considerations and choices must be made to tailor DFSMS to suit the various types of data sets and the individual customer's systems environment and organization.

In general, most of your data sets can be managed with DFSMS storage pools, thus reducing the workload and the interaction of your DB2 database administrators (DBAs) and storage administrators. Even the most critical data, as defined with service level agreements (SLAs) or as revealed by monitoring, can be managed with special attention.

#### Benefits of DFSMS

Using DFSMS, the DB2 administrator gains the following benefits:

- ▶ Simplified data allocation
- ▶ Improved allocation control
- ▶ Improved performance management
- ▶ Automated disk space management
- ▶ Improved data availability management
- ▶ Simplified data movement

See 4.1.3, "Benefits" on page 202 for more details.

Another important benefit is that, with DFSMS, the DB2 environment is positioned to take immediate advantage of available and future enhancements. For example, the following enhancements are available today to DB2 with the appropriate level of DFSMS:

#### ▶ DFSMS 1.5

z/OS consolidates adjacent extents for VSAM SMS-managed data sets when extending on the same volume. VSAM extent consolidation is automatic and requires no action on your part. Starting with z/OS 1.5, Linear data sets (LDSs) began to use extent consolidations. VSAM LDSs that are used for DB2 must be SMS-managed for extent consolidation to take effect.

#### ▶ DFSMS 1.6

Parallel access volume (PAV) is a new option. With Enterprise Storage Server® (ESS) and DS8000®, you can define alias device numbers to represent one physical device that allows for multiple I/O operations to be started at one time. With the new DFSMS storage class option of PAV, you can ensure that data sets that require high performance are allocated only to volumes that use the ESS PAV option. DB2 is using PAV to reduce I/O queuing in z/OS.

► DFSMS 1.7

DFSMS now supports sequential data sets that can exceed the previous size limit of 65535 tracks per volume. New data sets can be created as *large format data sets*, which are physical sequential data sets with the ability to grow beyond the previous size limit.

► DFSMS 1.8

The OAM DB2 *Binary Large Object Support* enables objects larger than 32 KB to be stored using DB2's large object (LOB) support and the binary large object (BLOB) data type. The LOB environment is intended to be used for objects larger than 32 KB continuing the need for our 4 KB and 32 KB tables

► DFSMS 1.10

*Extended address volumes* (EAV) adds support for volumes with a size greater than the previous limit of 65520 cylinders. An extended address volume is by definition 65521 cylinders or larger, up to a maximum of 262,668 cylinders in z/OS V1R10. DB2 can benefit from the relief of table spaces size constraints.

► DFSMS 1.11

The *enhanced striping volume selection* function has removed several restrictions. New support was added, including the use of volume selection preference attributes, so that striping volume selection now functions much more like conventional volume selection. Striping volume selection is used for SMS allocation or recall of a striped data set, that is, an extended format data set that occupies multiple volumes. DB2 can benefit from the new volume selection process by controlling the placement of striping data set in an automated way.

► DFSMS 1.12

DFSMS 1.12 increases the maximum volume size to 525,336 cylinders. Other new functions are as follows:

- DFSMSHsm fast replication, tape support
- DFSMSHsm fast replication, data set recovery
- PDSE 64-bit buffering, and caching data beyond closure of a PDSE
- Catalog enhancements
- New diagnostic tools
- A command to communicate with the device manager address space
- Creation of an ACDS from an SCDS
- Enhanced volume selection performance
- OAM enhancements
- SMS enhancements
- DFSMSrmm enhancements
- Tape data set authorization
- DFSMSHsm enhancements

DB2 for z/OS can benefit from z/OS startup time reduction through z/OS allocation, DFSMSdfp, and global resource serialization (GRS) improvements; address spaces opening up many thousands of data sets see more benefit.

DFSMS supports most additional data set types and data sets in EAVs. The EAVs can help relieve storage constraints, and can simplify storage management by providing the ability to manage fewer, large volumes rather than many small volumes.

For the most current information about the supported functions for DB2 and DFSMS, see “Related publications” on page 393.

## **Managing DB2 data sets with DFSMS**

The DB2 administrator can use DFSMS to achieve all the objectives for data set placement and design. DFSMS has the necessary flexibility to support everything the DB2 administrator might want. There is no reason for not taking advantage of DFSMS for DB2 data sets.

To achieve a successful implementation, an agreement between the storage administrator and the DB2 administrator is required so that they can together establish an environment that satisfies both their objectives.

## **Examples for managing DB2 data sets with DFSMS**

This book shows examples that describe one possible way to manage DB2 data sets with DFSMS. The examples can give an idea of the possibilities that DFSMS offers for DB2. Each example is only one out of many choices of how a medium to complex installation might approach the implementation of DB2 data sets with DFSMS.

Many installations might find a simpler implementation more adequate; others might want to have a more specific management than the one shown.

## **Data placement**

With smaller disk devices, without cache, data locality was previously important for performance, to reduce seek and rotation times. The new disk architectures, with redundant array of independent disks (RAID) architecture and with cache in the gigabyte sizes, have a noticeable effect on database physical design considerations. Conventional database design rules based on data set placement are becoming less important and can be ignored in most cases. Data placement can be done in automated way with DFSMS ACS routines.

## **Large cache**

Most storage servers with large cache (greater than 1 GB) ignore the bypass cache or inhibit cache load requests from the application. They always use the cache; however, they continue to take into account the specifications of usage from the applications by simply scaling down or up the track retention into the cache for reuse.

Installations that have these devices can use sequential caching as an installation option. Installations with a mixture of devices with large, small, or no cache can benefit from the bypass cache option.

## **SMS storage groups**

Volume separation is easy when you have hundreds of volumes available. However, this separation is good only if your volumes have separate access paths. Path separation is important to achieve high parallel data transfer rates.

Without DFSMS, the user is responsible for distributing DB2 data sets among disks. This process must be reviewed periodically, either when the workload changes, or when the storage server configuration changes.

With DFSMS, the user can distribute the DFSMS storage groups among storage servers with the purpose of optimizing access parallelism. Another purpose can be managing availability for disaster recovery planning. This purpose can be combined with the previous purpose by letting DFSMS automatically fill in these storage groups with data sets, by applying policies that are defined in the automatic class selection routines.

Changes to the topology of the storage group can be managed to minimize the application outages. This technique can be done simply by adding new volumes to the storage group, then managing the allocation ennoblement (opening it on new volumes, closing it on volumes

to be removed), and finally removing the volumes you want to exclude from the storage group. All those functions can be accomplished while the data is online. Data sets that were unmovable, never-closed, or never reallocated can be moved by using remote copy techniques, and then, after a short outage, the critical application can be switched onto the new volumes.

### **Performance management**

Monitoring I/O performance of DB2 requires teamwork between DB2 and storage administrators to adopt a common approach with tools of both disciplines in analyzing performance situations. Performance monitoring should be done at the storage group level to have a consistent action.

## **1.2.2 DB2 and storage servers**

DB2 has certain special requirements in the way its storage objects are defined and utilized. Disk technology has evolved introducing RAID architecture, large cache, Intelligent Write Caching (IWC), FlashCopy, Storage Pool Striping, Extended Address Volume, virtual architecture, Adaptive Multi-stream Prefetching (AMP), and Metro Mirror, Global Copy, Global Mirror, Metro/Global Mirror, z/OS Global Mirror, and z/OS Metro/Global Mirror. DBAs and storage administrators must agree on common actions before taking advantage of the available enhancements.

### **IBM Enterprise Storage Server**

The first IBM Enterprise Storage Server (ESS) was introduced in 1999 to meet the need for high performance, scalable, flexible, and available storage systems with advanced management capabilities. The first product in IBM Seascape architecture family of products (see “The Seascape architecture” on page 6), is the ESS, a SAN-ready disk storage system providing universal access across all major server platforms. It employs advanced hardware and software technologies to deliver breakthrough levels of performance and maximize data sharing throughout the enterprise.

An enhancement to the ESS, ESS Copy Services, provides three varieties of replication of mission-critical data. FlashCopy delivers near-instantaneous, nondisruptive, point-in-time copies within the ESS; Peer-to-Peer Remote Copy (PPRC) implements dynamic synchronous mirroring to a remote ESS; and in IBM System z environments, asynchronous copying to a remote site is possible using Extended Remote Copy (XRC).

The follow-on family to ESS is the DS8000 family. The IBM ESS is built on the Seascape architecture.

The new ESS can contribute to DB2 9 by delivering solutions in the performance and backup area through functions such as striping, virtualization of tape workload, ESS Copy Services (FlashCopy, PPRC, or XRC).

The Seascape architecture is the core to deliver such a solution to DB2.

### **The Seascape architecture**

Seascape is a blueprint for comprehensive storage solutions optimized for a connected world.

The Seascape architecture integrates leading technologies from IBM (including disk storage, tape storage, optical storage, powerful processors, and rich software function) to provide highly reliable, scalable, versatile, application-based storage solutions that span the range of servers from PCs to supercomputers.

At its heart, Seascope architecture uses an open, industry-standard storage server that can scale up exponentially in both power and performance. Because the storage servers can integrate snap-in building blocks and software upgrades, you can quickly deploy new or improved applications and rapidly accommodate new data and media types. In this way, Seascope storage servers become an essential element for storing, manipulating, and sharing data across the network.

Figure 1-1 shows how the Seascope architecture takes place in the current DS8000.

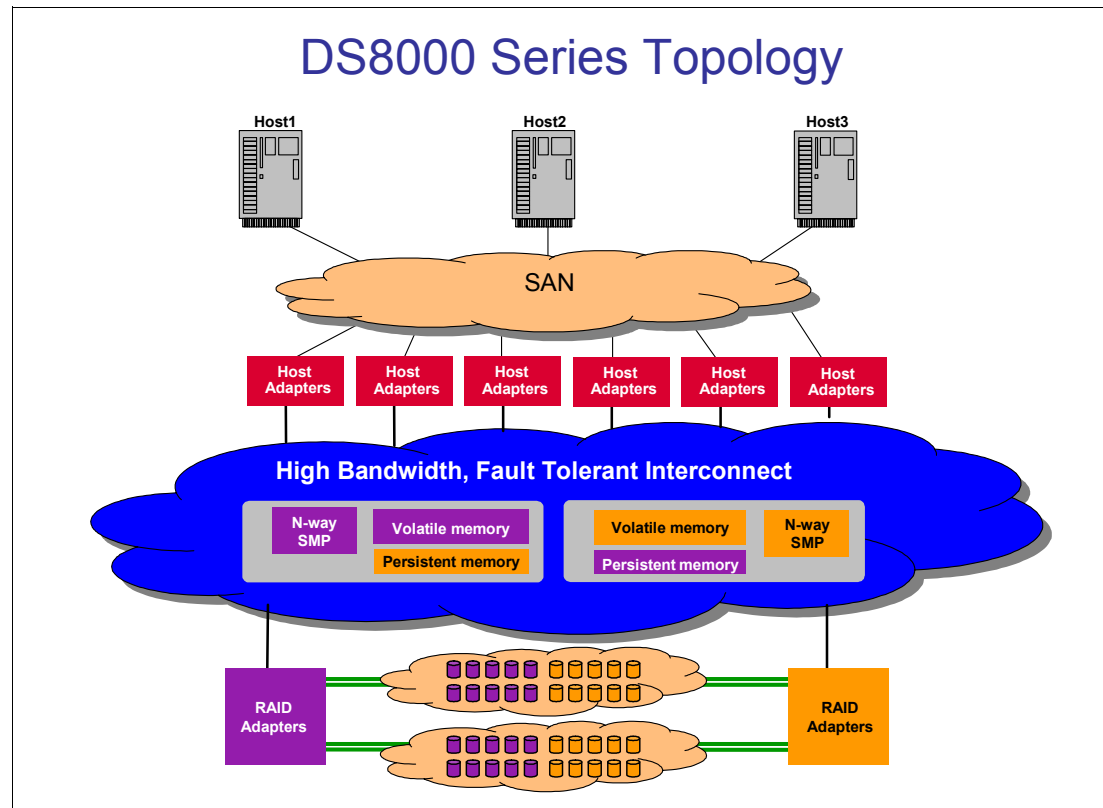


Figure 1-1 DS8000 and Seascope architecture

The Seascope architecture is the key to the development of IBM storage products. Seascope allows IBM to take the best of the technologies developed by the many IBM laboratories and integrate them, producing flexible and upgradeable storage solutions. This Seascope architecture design has allowed the IBM TotalStorage® Enterprise Storage Server to evolve from the initial E models to the succeeding F models, 800 models, and to the recent DS8000 family. Each features new, more powerful hardware and functional enhancements and is always integrated under the same successful architecture with which the ESS was originally conceived.

## IBM enterprise tape systems

Backup and restore is the most simple and basic solution to protect and recover your data from failure by creating another copy of data from the production system. Use the second copy of data to restore that data to the time of the data backup. Backup is a daily IT operation task where production, application, systems, and user data are copied to a separate data storage media, in case they are needed for restore operations. Over the past few years, the growth in the demand for data storage and reliable backup and archiving solutions has greatly increased the need to provide manageable and cost-effective tape library products. The value

of using tape for backup purposes has only gradually become obvious and important in these environments.

System administrators want technologies that enable them to efficiently and economically manage the explosive growth in stored data. As the amount of data increases, the backup process takes longer.

Tape storage can provide effective solutions to backups and archives residing on disk. Tape storage also helps maintain data availability, reduce storage costs, and provide disaster recovery functionality.

IBM enterprise class tape products offer high performance, availability, reliability, and capacity to help meet the needs of customers seeking enterprise class storage solutions for mass storage, data archiving, enterprise backup, and disaster recovery. IBM enterprise class storage products provide on-demand solutions for business.

### **IBM TotalStorage and System Storage tape drives**

This section explains the terminology used in the storage tape space. The following devices are equipped with tape drives to provide real tape storage media:

- ▶ The Automated Tape Library (ATL) is a device consisting of robotic components, cartridge storage areas, tape subsystems, and controlling hardware and software, together with the set of tape volumes that reside in the library and can be mounted on the library tape drives. ATLs do not provide tape virtualization or automatic tape stacking.
- ▶ The Virtual Tape Server (VTS) is a hardware-based solution that addresses both tape cartridge use and tape device use, and therefore overall tape subsystem costs. The VTS was introduced and made available in 1996 as a new type of storage solution, combining a high-speed disk cache with tape automation, tape drives, and intelligent storage management software running on a server.
- ▶ The IBM System Storage® Virtualization Engine TS7700 is the newest member of the IBM TS7000 Virtualization Family. It represents the fourth generation of IBM Tape Virtualization for mainframe systems and replaces the highly successful IBM TotalStorage Virtual Tape Server.

The System Storage Virtualization Engine is transparent to applications such as DB2 9. Although VTS and TS7700 each have a disk front-end that eventually writes to physical tape, both VTS and the TS7700 system are considered as tape devices by MVS and return the device type back to DB2 as tape.

For DB2 9 VTS and TS7700 system are still considered tape devices. For recovery serialization implications with tapes, see *DB2 9 for z/OS: Backup and Recovery I/O Related Performance Considerations*, REDP-4452.

## TS7700 Virtualization Engine Operation



The benefits are as follows:

- ### ***Tape for DB2 data***

- ▶ HSM-migrated archive logs
- ▶ Image copies
- ▶ Very large sortwork data sets
- ▶ Other assorted less frequently used data sets

Similar considerations apply for RECOVER jobs with image copies and rolling back or aborting of a long-running unit of work.







## Disk and tape environment overview

In this chapter, we consider the disk architecture from a DB2 point of view. We focus on general concepts and suggestions for their practical implementation, rather than on technical details. However, to be able to take full advantage of what the new disk and tape environment can deliver to DB2, you need to understand the recent enhancements of disk and tape I/O architecture. Therefore, we explain several technical details to help facilitate the mutual understanding of the most important storage terms between DB2 administrators and storage administrators. Several considerations in this chapter also apply to the new tape server environments, such as the IBM Virtualization Engine TS7700.

This chapter contains the following topics:

- ▶ How the new storage server technologies interact
- ▶ Evolution of storage subsystems architecture
- ▶ Disk capacity
- ▶ zArchitecture data scalability
- ▶ Redundant Array of Independent Disks (RAID)
- ▶ HyperSwap
- ▶ Seascape architecture
- ▶ Disk virtualization
- ▶ Caching algorithms in DS8000
- ▶ Common performance considerations for System z
- ▶ Copy Services
- ▶ Solid-state drives with DS8000
- ▶ Tapes: Storage Tier 3 in z/OS environment
- ▶ Partitioned data sets
- ▶ New technologies and functions benefit DB2 for z/OS

## 2.1 How the new storage server technologies interact

In this chapter, we describe storage servers' new technologies and functions for one reason: to take advantage of those new technologies and functions to increase the performance and throughput of the DB2 workloads.

We do not mean DB2 performance and throughput increase by optimizing SQL queries but rather how we can take advantage of the performance and throughput to get the needed data stored outside of the CPU for DB2 processing. This process starts at the host (CPU) and ends on the hard disk drive (HDD). Therefore, we have to take care of all components involved into this process: the z/OS access method going over the z/OS IOS (Input/Output Supervisor) component, CPU channel, channel path (FICON®), direct access storage device (DASD) storage subsystem (DS8000), host adapter, DS8000 virtualization level with cache and HDD mapping, and down to the real spinning disk drive.

Adjusting one component in the flow does not make sense. Rather, we need to control all involved areas and take care that all components are adjusted to provide the right level of service to each other.

Striping is an effective way of increasing sequential throughput by spreading VSAM control intervals across multiple devices.

At the beginning, when striping arrived at the data access space we had only the possibility to do data striping with z/OS (DFSMS) data access methods. Striping spreads the data across multiple DASDs (VOLSER). This data striping method is still valid because it begins at the z/OS access method level, that is the striping take place before the data is going over the channel to the DASD storage subsystem. We have the benefit of spreading data over more than one channel. See 2.9.9, "Data striping by access method" on page 57 for details about how it works.

With the introduction of the RAID, since Enterprise Storage Server (ESS) technology, we had striping on the disk storage subsystem level too. For how RAID is implemented, how it works, and what its benefits are, see 2.5, "Redundant Array of Independent Disks (RAID)" on page 35.

The virtualization of DASD devices allows the next stage of striping. The DS8000 storage subsystem manages all DASD space in extent pools and the storage administrator has the possibility to define *Storage Pool Striping*. This DS8000 option does a DASD (VOLSER) striping across extent pools. For how the Storage Pool striping is implemented and how it works, see section 2.9.6, "Extent pools" on page 51 and "Storage Pool Striping: rotate extents" on page 55.

Solid-state drive (SSD) technology, together with the Easy Tier function, can help provide relief to hot-spot constraints. For details, see 2.13, "Solid-state drives with DS8000" on page 102.

The next important technology step in this domain is the cache. Although the functionality of cache itself has not changed, how it works has changed significantly. All the known parameters such as MUST CACHE, CACHE BYBASS, DASD FAST WRITE, and others, are more or less obsolete. In the past, although you needed to manage the cache with these parameters, today the DS8000 cache is always involved in the data transfer process down to the hard disk drive. Even more, DS8000 has introduced new high sophisticated caching algorithms to manage the cache. No longer is cache managed from the outside. The DS8000 caches everything. No data goes to hard disk drives without first going through the cache. The DS8000 cache management analyzes the data access pattern and the read/write commands, and then determines how the data should be cached based on the data access

pattern and read/write commands frequency. For information about how the DS8000 cache management is implemented and works, see 2.10, “Caching algorithms in DS8000” on page 64.

Finally, the channel section is next on the way to the hard disk drive. In recent years, this area has improved greatly. Both the hardware itself has improved, with the latest FICON Express8 (8 Gbps) technology, and also channel data transfer protocol and how z/OS addresses the DASD volume itself has improved. Each I/O request is finally tied to an I/O represented by a unit control block (UCB) in z/OS. If there are several requests against a UCB, the requests are queued. When a constraint relief is provided at the DASD storage subsystem side, we have to resolve a corresponding serious constraint issue at the channel side. To relieve this constraint, z/OS introduced the parallel access volume (PAV)/Hyper parallel access volume (HyperPAV) function. PAV allows multiple UCBs representing the same DASD volume. In a parallel sysplex environment, the Workload Manager (WLM) dynamically manages the HyperPAV-defined device. z/OS also introduced multiple allegiance, which supports concurrent I/O requests from multiple systems to be active against the same logical volume, if they do not conflict with each other. For how PAV/HyperPAV, WLM, multiple allegiance, and I/O priority queuing work together and are implemented, see 2.4.2, “Parallel access volume (PAV)” on page 25, 2.4.4, “HyperPAV” on page 29, 2.4.4, “HyperPAV” on page 29, and 2.4.5, “I/O priority queuing” on page 32.

Channel protocol is not the same as it was ten years ago. By introducing the MIDAW facility and High Performance FICON for system z (zHPF), a combination of a new channel command word process and a new channel protocol took place. Both new functions increase the bandwidth between the actual DASD storage subsystem DS8000 and the CPU. See 2.11.1, “Host connections to System z servers” on page 68 for details.

The capacity increase allows going well beyond the DASD volume size of a 3390 Model 9. The new DASD volume, called *extended address volume (EAV)*, is a volume with more than 65,520 cylinders. An EAV increases the amount of addressable DASD storage per volume beyond 65,520 cylinders by changing how tracks are addressed on the volumes. The extended address volume is the next step in providing larger volumes for z/OS. An EAV volume is defined as a 3390 Model A in z/OS. The benefit of this support is that the amount of z/OS addressable disk storage is further significantly increased. Users who are approaching the four-digit device number limit, can allocate data on the EAV DASD up to a maximum of 262,668 cylinders in z/OS V1R10. Customer applications using large VSAM data sets, such as those often used by DB2, faced the four-digit device limit. To migrate easily to an EAV, you have the function of *dynamic volume expansion (DVE)*. This function simplifies data growth by allowing volume expansion without taking volumes offline. Using DVE significantly reduces the complexity of migrating to larger volumes. See 2.4.6, “Dynamic volume expansion” on page 33 and 2.4.6, “Dynamic volume expansion” on page 33.

As a DBA, you are familiar with how z/OS access method (VSAM striping), RAID, Storage Pools Striping, SDD, and Easy Tier can influence the DB2 performance and throughput depending on the workload pattern. Even if there is a high level of virtualization in the DS8000 there will be still situations where you as a DBA should understand what the storage administrator can do for you, such as how to balance DB2 data sets between two DS8000 Storage Facility Image 1 (SFI 1) and Storage Facility Image 2 (SFI 2). For details, see 2.9.1, “Storage system virtualization” on page 46.

**Note:** Do not adjust only one component in the system. Always consider the whole system because the components work together as a team.

A description of the enhancements in the previous sections does not divulge how to manage the data. The data is the capital asset of the enterprise and therefore must be always

retrievable. This huge amount of data stored in data storage subsystems must be managed so that the data is always retrievable.

Having all enterprise data always retrievable is not feasible. The first step must be a qualification of the enterprise data. The customer must define a priority list about the enterprise data. Considerations include which data is the most important, and how fast the enterprise data must be restored after a breakdown of a DASD volume, a storage subsystem, or maybe a disaster.

Depending on the priority, we can choose and set up the appropriate Copy Service provided by the DS8000. For a breakdown of an HDD we can use RAID 6 or RAID 10. Which RAID level you use depends on your need. For more details about the RAID functionality and how it works see 2.5, “Redundant Array of Independent Disks (RAID)” on page 35. This function is not a DS8000 Copy Service but it is the first level of enterprise data availability. Which RAID level you choose depends on your requirements.

Copy services delivered by ESS and DS8000 are as follows:

- ▶ Extended remote copy (XRC), see “Extended remote copy (XRC)” on page 80
- ▶ Peer-to-Peer Remote Copy (PPRC), see “Peer-to-peer remote copy (PPRC)” on page 79
- ▶ PPRC-XD (Peer-to-Peer Remote Copy Extended Distance), see “Peer-to-peer remote copy extended distance (PPRC-XD)” on page 79
- ▶ FlashCopy, depending of the storage subsystem you are using; for ESS, see “FlashCopy” on page 81; for DS8000, see “FlashCopy and IBM FlashCopy SE” on page 84 and “FlashCopy options” on page 87
- ▶ Incremental FlashCopy, see “Incremental FlashCopy” on page 87
- ▶ Consistency Group FlashCopy, see “Consistency Group FlashCopy” on page 89
- ▶ Persistent FlashCopy, see “Persistent FlashCopy” on page 91
- ▶ Remote Pair FlashCopy, see “Remote Pair FlashCopy” on page 91
- ▶ Remote Mirror and Copy, see “Remote Mirror and Copy” on page 93:
  - Metro Mirror, see “Metro Mirror” on page 93
  - Global Copy, see “Global Copy” on page 94
  - Global Mirror, see “Global Mirror” on page 95
  - Metro/Global Mirror, see “Metro Global Mirror” on page 97
  - z/OS Metro Global Mirror, see “z/OS Global Mirror” on page 100

At the end of the DS8000 Copy Service, a summary lists all copy services available with possible combinations of copies. See “DS8000 Copy Services matrix” on page 82.

After talking about the DASD storage subsystems, we now look at the last tier in the data placement hierarchy, the tape.

A similar evolution exists in the tape space as in the disk space. A tremendous move to virtualization took place, from stand-alone tape drives to tapes installed in an automated tape library (ATL) and to a 7700 Virtualization Engine.

When a tape data set is written to a tape drive, this can be a native tape drive such as the TS1130 tape drive or it can be a virtual tape drive in the TS7700 Virtualization Engine. The TS7720 models are not attached to physical tape drives and only emulate virtual tape drives to the host. The tape data is actually written to disk. The TS7740 models also emulate virtual tape drives to the host but are also attached to physical backend tape drive residing a tape library.

However, those physical tape storage sizes of 200 MB (3480) increased to 300,000 MB (3592) and 1 TB (TS1130), and a data transfer rate up to 350 MBps with 3:1 compression. See section 2.14.1, “Tape virtualization” on page 106 for how virtualization is implemented and how it works.

Similar terminology as in the DASD space are the components tape volume cache (TVC), physical drives, stacked volumes, virtual drives, virtual volume, and logical volumes. “Tape virtualization design” on page 108 shows how the components work together.

The tape space still has the compression function. The data is compressed when written to the TVC.

For a detailed description of how to satisfy the need to always provide access to the enterprise data at a reasonable cost, see 2.14.2, “TS7700 Virtualization Engine” on page 109. The TS7700 can work as a grid installation where the separate TS7700s (up to three) are physically dispersed. See “Grid configuration” on page 110.

The TS7700 supports tape encryption, secure-data erase, copy export, and Multi Cluster Grid configurations for high availability and multi-site support.

## 2.2 Evolution of storage subsystems architecture

When we look at storage subsystems today we can see the major change has been moving from a physical mapped infrastructure towards a virtual mapped infrastructure. In the past, a hard-implemented path traveled from the data processing unit to the spinning magnetic disk: a dedicated connection existed until the read or write task was completed. This configuration is no longer valid. Today, enterprise storage subsystems take over the responsibility for the upcoming read or write but we do not really know how the data is handled. The evolution of the disk architecture has progressively separated the concept of volume from the concept of physical device. The storage subsystem today is more of a processing system itself than simply a control unit with magnetic disk. This change allows delegating more functionality to the enterprises storage subsystem; DB2 can take the advantage of these new functions.

Another consideration is related to the central processors that support massive real storage, in the order of terabytes, faster CPUs with more parallelism, and increase in the bandwidth from ESCON® to FICON and zHPF. Although, the hard disk drives have not improved substantially in speed in the last ten years, they have become larger and provide increased availability.

Users have a choice in deciding volume capacity, separate levels of RAID protection, and the use of the emerging SSD technology.

## 2.3 Disk capacity

Figure 2-1 on page 16 shows various IBM DASD device types. In the 1980s, 3380 devices were used. Capacity went from 885 to 2,655 cylinders per volume. When storage density increased, new device types were introduced at the end of the 1980s. Those types were called 3390. Capacity per volume ranged from 1,113 to 3,339 cylinders. A special device type, model 3390-9, was introduced to store large amounts of data that needed fast access. The track geometry within one device category was (and is) always the same; this geometry means that 3380 volumes have 47,476 bytes per track, and 3390 volumes have 56,664 bytes per track.

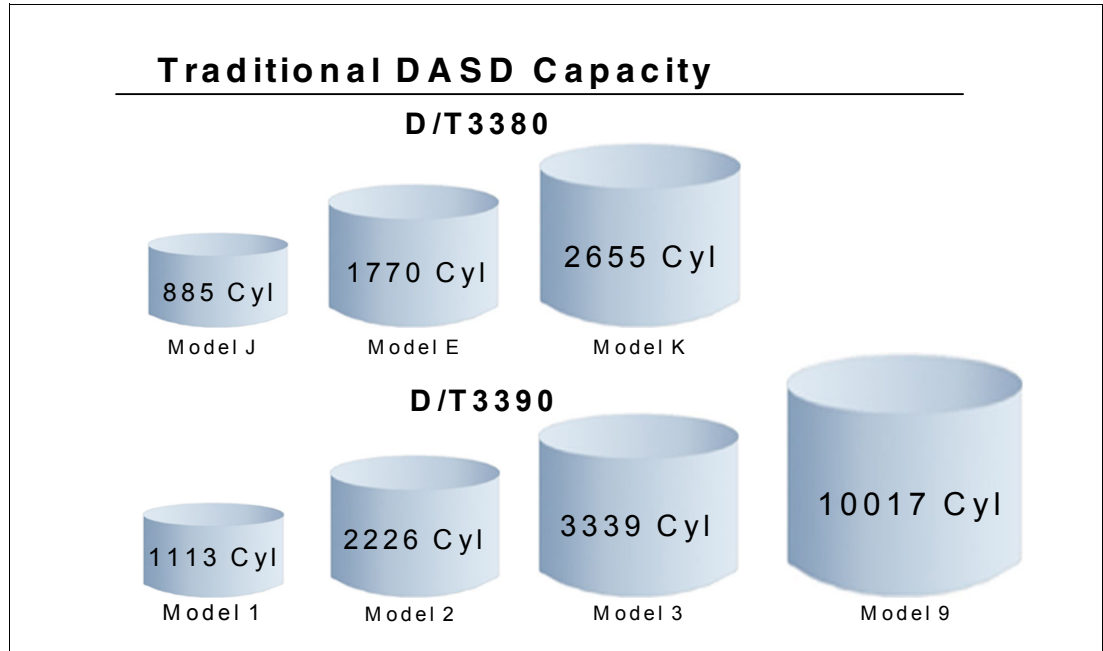


Figure 2-1 Traditional DASD capacity

### 2.3.1 Large volumes before z/OS V1R10

Large volumes 3390-27 and 3390-54 were the first two steps on the way to relieve the constrained volume size of 10017 cylinders of the 3390-9.

The IBM TotalStorage Enterprise Storage Server (ESS) initially supported custom volumes of up to 10017 cylinders, the size of the largest standard volume, the 3390 model 9. This limit was set by the operating system software. The IBM TotalStorage ESS large-volume support enhancement of 2001, increased the upper limit respectively to 32760 cylinders (approximately 27 GB) and 65520 cylinders (approximately 54 GB). The enhancement is provided as a combination of IBM TotalStorage ESS Licensed Internal Code changes and system software changes, available for z/OS and z/VM®.

The IBM Enterprise Storage Server emulates the IBM 3390. On an emulated disk or on an IBM Virtual Machine (now z/VM) minidisk, the number of cylinders per volume is a configuration option. It might be less than or greater than the stated number. If so, the number of bytes per device will differ accordingly. The IBM ESS Model 1750 supports up to 32760 cylinders and the IBM ESS Model 2107 supports up to 65520 cylinders. See Figure 2-2 on page 17.

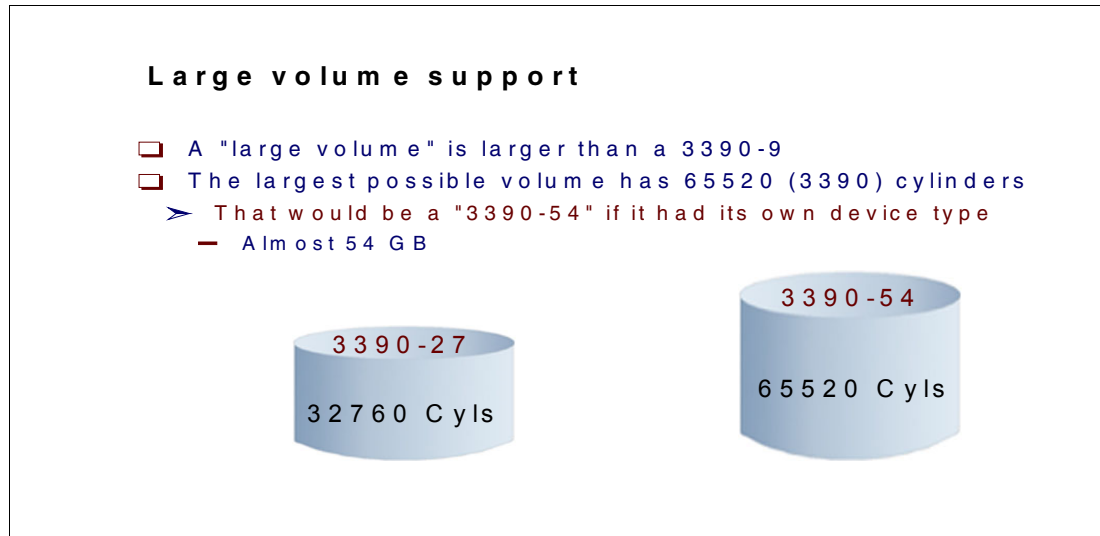


Figure 2-2 Large volume support

Large volume support is available on z/OS operating systems, the ICKDSF, and DFSORT utilities.

Large volume support must be installed on all systems in a sysplex prior to sharing data sets on large volumes. Shared system and application data sets cannot be placed on large volumes until all system images in a sysplex have large volume support installed.

## Easy Tier

Easy Tier adds another layer of storage virtualization. Easy Tier helps take advantage of the performance and cost of fast but expensive SSD devices. The extra level of virtualization that it provides transfers more responsibility of space management to the system and relieves you from that responsibility. Easy Tier can dynamically move a logical disk address to optimize performance. With Easy Tier, the control unit determines the cache-miss frequency of each *extent*, an extent being a chunk of 1113 cylinders. If a DS8000 storage pool contains a mixture of slow disks (such as HDDs) and fast disks (such as SSDs), the DS8000 tries to minimize HDD cache misses by placing the busiest and most cache-unfriendly extents on the fastest disks.

## Large volumes support considerations

Benefits of large volumes can be briefly summarized as follows:

- ▶ They reduce management tasks by allowing you or the system to manage a smaller number of volumes.
- ▶ They reduce the number of multivolume data sets that you or the system has to manage.
- ▶ They allow you to address more data than within the existing 64K address number limit.
- ▶ They allow you to enable more addresses to be used as aliases, which is good for minimizing UCB contention.

Every control unit model manages space differently, but the DS8000 family manages space in units of what it calls *extents*. Each extent in a 3390 pool consists of 1113 cylinders, no matter what the volume size is. Each volume consists of some multiple of extents. Although you may define any EAV volume size that you want, if the volume size is not an even multiple of 1113 cylinders, you waste space.

Because of storage virtualization in the control unit, volume boundaries bear little relationship to physical disks. This is especially true of Easy Tier, but it has been true since the invention of RAID architecture. That relationship depends on how the disks are configured inside the control unit. Two concurrent I/Os to separate volumes may access the same disk, and two concurrent I/Os to the same volume may access separate disks. However, one fact is always true: all volumes in the same logical control unit (LCU) share the same pool of UCBs. By minimizing the number of volumes in an LCU, you can maximize the number of aliases in the LCU. That minimizes UCB contention. Furthermore, if each LCU uses a separate pool of physical disks, you can be sure that two LCUs will never contend for the same physical disks, in which case you can think of LCUs as the smallest granularity of space to manage. What happens within the LCU is managed by the system, and is out of your control.

The benefits of large volumes must be weighed against a couple of possible disadvantages. For example, a physical volume dump can take longer. Also, because reserves serialize access to a volume, large volumes might increase reserve contention. Usually these disadvantages are negligible for DB2 databases, but be cautious for volumes that are used as scratch pools because data set allocation and deletion might cause reserve contention.

### 2.3.2 Extended address volume (EAV)

An EAV is a volume with more than 65,520 cylinders. An EAV increases the amount of addressable DASD storage per volume beyond 65,520 cylinders by changing how tracks on volumes are addressed. The EAV is the next step in providing larger volumes for z/OS. z/OS provided this support first in *z/OS V1R10* of the operating system. Over the years, volumes have grown by increasing the number of cylinders and thus gigabyte capacity. However, the existing track addressing architecture has limited the required growth to relatively small gigabyte capacity volumes, which has put pressure on the four-digit device number limit. Previously, the largest available volume is one with 65,520 cylinders or approximately 54 GB. Access to the volumes includes the use of PAV, HyperPAV, and FlashCopy space-efficient volumes feature<sup>1</sup>

With EAV volumes, an architecture is implemented that provides a capacity of hundreds of terabytes for a single volume. However, the first releases are limited to a volume with 223 GB or up to 262,668 cylinders. For reference information, see *DFSMS V1.10 and EAV Technical Guide*, SG24-7617.

#### 3390 Model A

A 223 GB volume has to be configured in the DS8000 as a 3390 Model A. However, a 3390 Model A is not always an EAV. A 3390 Model A is any device configured in the DS8000 to have more than 65,220 cylinders. Figure 2-3 on page 19 illustrates the 3390 device types.

**Note:** With the 3390 Model A, the model A refers to the model configured in the DS8000. It has no association with the 3390A notation in HCD that indicates a PAV-alias UCB in the z/OS operating system. The Model “A” was chosen so that it did not imply a particular device size as previous models 3390-3 and 3390-9 did.

---

<sup>1</sup> The FlashCopy SE feature (not used by DB2) is described in “IBM FlashCopy SE options” on page 92.



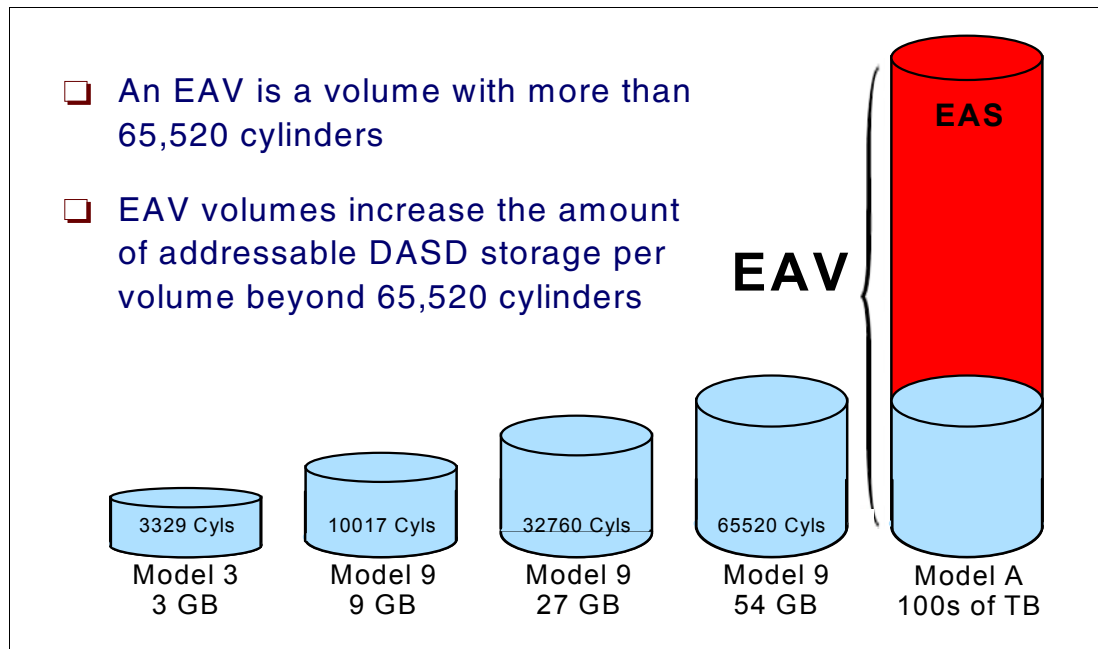


Figure 2-3 EAV Volumes

Note the following points regarding extended address volumes:

- ▶ Only 3390 Model A devices can be EAV.
- ▶ EAV is supported by z/OS V1R10 and later releases.
- ▶ The size is limited to 223 GB (262,668 cylinders) in z/OS V1R10 and V1R11

**Important:** The 3390 Model A as a device can be configured to have 1 - 268,434,453 cylinders on an IBM DS8000. It becomes an EAV if it has more than 65,520 cylinders that are defined. With current z/OS releases, this maximum size is not supported.

## EAV benefits

The following list summarizes the benefits of EAV:

- ▶ Offers increased z/OS addressable disk storage
- ▶ Provides constraint relief for applications using large VSAM data sets, extended-format data sets.
- ▶ For 3390 Model A, devices can be configured to have from 1 to 268,434,453 cylinders (architectural maximum).
- ▶ Is managed by the system as a general purpose volume.
- ▶ Works well for applications with large files. PAV and HyperPAV technologies help by allowing I/O rates to scale as a volume gets larger.

With this support, the amount of z/OS addressable disk storage is further significantly increased. This increase provides relief for users that are approaching the four-digit device-number limit by providing constraint relief for applications using large VSAM data sets, such as those used by DB2, CICS®, zFS file systems, SMP/E CSI data sets, and NFS-mounted data sets. This support is provided in z/OS V1R10 and enhanced in z/OS V1R11.

### 2.3.3 Data sets eligible for EAV volumes

EAS-eligible data sets are defined to be those that can be allocated in the *extended addressing space (EAS)*. This is the area on an EAV that is located above the first 65,520 cylinders. This area is sometimes referred to as *cylinder-managed space*.

In z/OS V1R10, the following VSAM data sets are EAS-eligible:

- ▶ All VSAM data types (KSDS, RRDS, ESDS, and LDS), which covers DB2, IMS, CICS, zFS and NFS:
  - SMS-managed and non-SMS-managed VSAM data sets
  - VSAM data sets on an EAV that were inherited from a prior physical migration or copy

With z/OS V1R11, extended-format sequential data sets are SMS-managed.

The following data sets can be allocated in the *base addressing space* of an EAV:

- ▶ SMS-managed VSAM (all types)
- ▶ Non-SMS VSAM (all types: KSDS, RRDS, ESDS, LDS)
- ▶ zFS data sets (which are VSAM LS)
  - zFS aggregates are supported in an EAV environment.
  - zFS aggregates or file systems can reside in track-managed space or cylinder-managed space (subject to any limitations that DFSMS has).
  - zFS still has an architectural limit of 4 TB for the maximum size of a zFS aggregate.
- ▶ Database (DB2, IMS) use of VSAM
- ▶ VSAM data sets inherited from prior physical migrations or copies
- ▶ With z/OS V1R11: Extended-format sequential data sets that are SMS-managed.

An EAS-ineligible data set is a data set that may exist on an EAV but is not eligible to have extents (through creating or extending) in the cylinder-managed space. The exceptions to EAS eligibility are as follows:

- ▶ Catalogs (basic catalog structure, or BCS) and VSAM volume data set (VVDS)
- ▶ VTOC (continues to be restricted to within the first 64K-1 tracks)
- ▶ VTOC index
- ▶ Page data sets
- ▶ VSAM data sets with imbed or keyrange attributes
- ▶ VSAM data sets with incompatible control area (CA) sizes

**Note:** In a future release, various of these data sets might become EAS-eligible. All data set types, even those listed here, can be allocated in the track-managed space on a device with cylinder-managed space on an EAV volume. Eligible EAS data sets can be created and extended anywhere on an EAV. Data sets that are not eligible for EAS processing can only be created or extended in the track-managed portions of the volume.

### 2.3.4 EAV volumes and multicylinder units

A *multicylinder unit (MCU)* is a fixed unit of disk space that is larger than a cylinder. Currently, on an EAV volume, a multicylinder unit is 21 cylinders and the number of the first cylinder in each multicylinder unit is a multiple of 21. Figure 2-4 illustrates the EAV and multicylinder units.

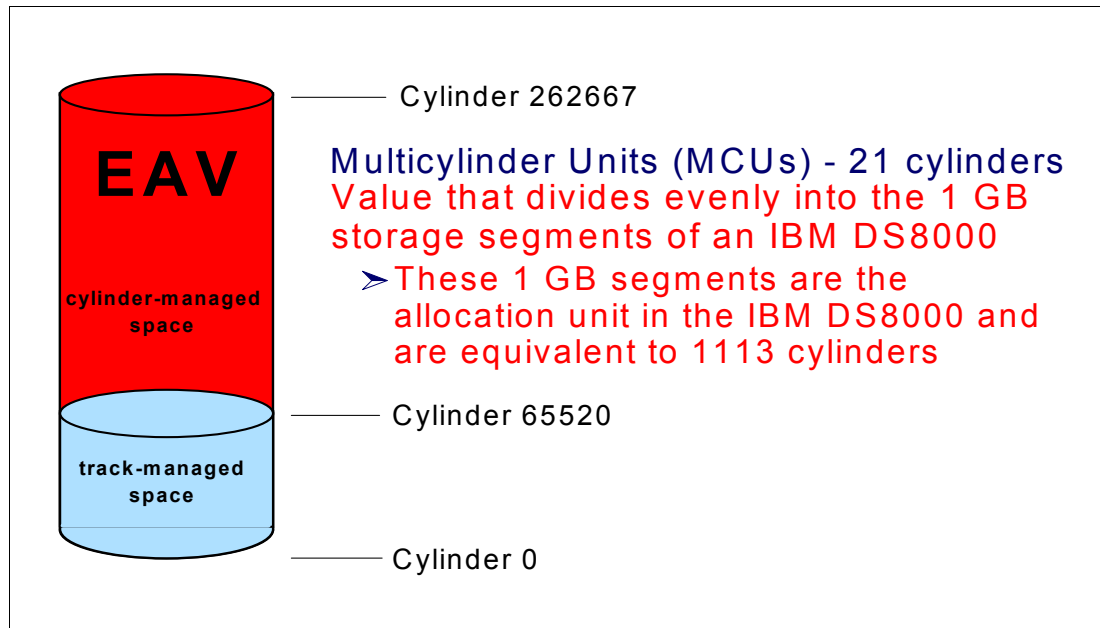


Figure 2-4 EAV and multicylinder units

As previously mentioned, the extended addressing space (EAS) is sometimes referred to as cylinder-managed space because it is space on the volume that is managed only in multicylinder units. Cylinder-managed space begins at cylinder address 65,520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space are rounded up to the next multicylinder unit. The cylinder-managed space exists only on EAV volumes.

The 21-cylinder value for the MCU is derived from being the smallest unit that can map out the largest possible EAV volume and stay within the index architecture with a block size of 8,192 bytes, as follows:

- ▶ It is also a value that divides evenly into the 1 GB storage segments of an IBM DS8000.
- ▶ These 1 GB segments are the allocation unit in the IBM DS8000 and are equivalent to 1,113 cylinders.
- ▶ These segments are allocated in multiples of 1,113 cylinders starting at cylinder 65,520.

One of the more important EAV design points is that IBM maintains its commitment to customers that the 3390 track format and image size, and tracks per cylinders, will remain the same as for previous 3390 model devices. An application using data sets on an EAV will be comparable to how it runs today on 3390 *numerics* kinds of models. The extended address volume has two managed spaces, as shown in Figure 2-4:

- ▶ Track-managed space

The track-managed space is the space on a volume that is managed in track and cylinder increments. All volumes today have track-managed space. Track-managed space ends at cylinder number 65519. Each data set occupies an integral multiple of tracks. Each data

set occupies an integral multiple of tracks. The track-managed space allows existing programs and physical migration products to continue to work. Physical copies can be done from a non-EAV to an EAV and have those data sets accessible.

► **Cylinder-managed space**

The cylinder-managed space is the space on the volume that is managed only in MCUs. Cylinder-managed space begins at cylinder address 65520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space are rounded up to the next multicylinder unit. The cylinder-managed space exists only on EAV volumes. A data set allocated in cylinder-managed space may have its requested space quantity rounded up to the next MCU.

Data sets allocated in cylinder-managed space are described with a new type of data set control blocks (DSCB) in the VTOC. Tracks allocated in this space will also be addressed using the new track address. Existing programs that are not changed will not recognize these new DSCBs and therefore will be protected from seeing how the tracks in cylinder-managed space are addressed.

The track-managed portion, which is one quarter of the volume, is the same size as a mod 54. The cylinder-managed portion is the remaining three quarters of the volume.

### **VTOC Index approach on EAV**

Free space is reported for both the track-managed portion of the EAV and another entry for the total amount of free space: cylinder-managed and track-managed.

The free space is determined by the system and kept in the VTOC INDEX (this is a new feature in DFSMS V1.10). Ensure that EAV volumes have an active index and system management procedures are in place to detect and rebuild an index if it gets disabled.

## **2.4 zArchitecture data scalability**

In the past decade, as processing power has dramatically increased, appropriate solutions have been deployed so that the amount of data that is directly accessible can be kept proportionally equivalent. Over the years DASD volumes have increased in size by increasing the number of cylinders and thus GB capacity.

However, the existing track-addressing architecture has limited growth to relatively small gigabyte capacity volumes. This architecture has placed increasing strain on the four-digit device number limit and the number of UCBs that can be defined. The largest available volume is one with 65,520 cylinders or approximately 54 GB, as shown in Figure 2-2 on page 17.

Rapid data growth on the z/OS platform is leading to a critical problem for various clients, with a 37% compound rate of disk storage growth between 1996 and 2007. The result is a real constraint on growing data on z/OS and the belonging subsystems like DB2. Business resilience solutions (GDPS®, HyperSwap®, and PPRC) that provide continuous availability are also driving this constraint.

### **Serialization granularity**

Since the 1960s, shared DASD can be serialized through a sequence of RESERVE/RELEASE channel command words (CCWs) that are today under the control of GRS, as shown in Figure 2-5 on page 23.

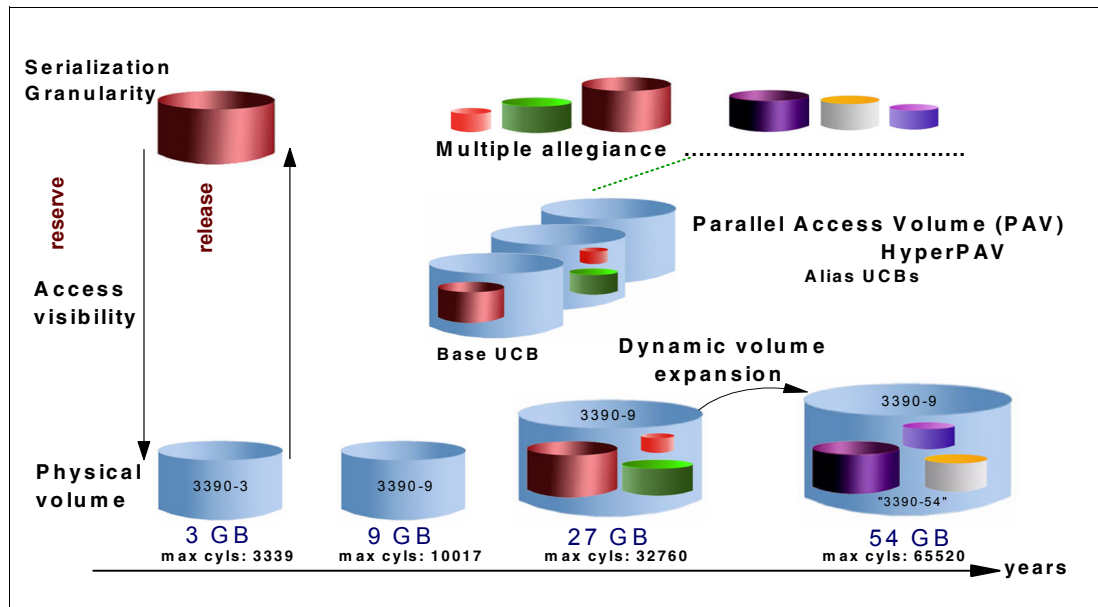


Figure 2-5 zArchitecture data scalability

This mechanism was useful if the volume of data so serialized (the granularity) was not too great. But when such a device grew to contain too much data, bottlenecks became an issue.

## DASD virtual visibility

Traditional S/390® architecture did not allow more than one I/O operation to the same S/390 device because such devices can only handle, physically, one I/O operation at a time. However, in modern DASD subsystems such as ESS, and DS8000, the device (such as a 3390) is only a logical view. The contents of this logical device are spread in head disk assembly (HDA) RAID arrays and in caches. Therefore, it is technically possible to have more than one I/O operation towards the same logical device. Changes have been made in z/OS (in IOS code), in the channel subsystem (SAP), and in ESS, and DS8000 to allow more than one I/O operation on the same logical device. This is called *parallel I/O*, and it is made possible by two functions:

### ► Multiple allegiance (MA)

MA provides the capability to support I/O requests from multiple systems, one per system, to be concurrently active against the same logical volume, if they do not conflict with each other. Multiple allegiance take place especially in Parallel Sysplex® environment, where RESERVE/RELEASE is on a higher level of serialization, such as an entire process that does many I/Os, like a DSS volume dump. This function is provided by default in the hardware.

See 2.4.1, “Multiple allegiance” on page 24 for more detail.

### ► Parallel access volume (PAV)

PAV allows concurrent I/Os to originate from the same z/OS image. Using PAV can provide significant performance enhancements in IBM System z environments by enabling simultaneous processing for multiple I/O operations to the same logical volume. PAV was introduced in z/OS V1R6 as a new option that allows specifying PAV capability as one of the volume selection criteria for SMS-managed data sets that is assigned to a storage class. Initially provided with the option of static and dynamic PAV, it also includes the HyperPAV feature.

PAV must be enabled by the MVS system programmer.

See 2.4.2, “Parallel access volume (PAV)” on page 25 to learn how PAV works and the benefit of the various types of PAV. See also 2.4.4, “HyperPAV” on page 29.

## 2.4.1 Multiple allegiance

Normally, if any System z host image (server or LPAR) does an I/O request to a device address for which the storage disk subsystem is already processing an I/O that came from another System z host image, the storage disk subsystem sends back a *device busy* indication. This method delays the new request and adds to processor and channel overhead. See Figure 2-6. This delay is shown in the Resource Measurement Facility™ (RMF™) activity report under the Pend time column. See Example 6-1 on page 326.

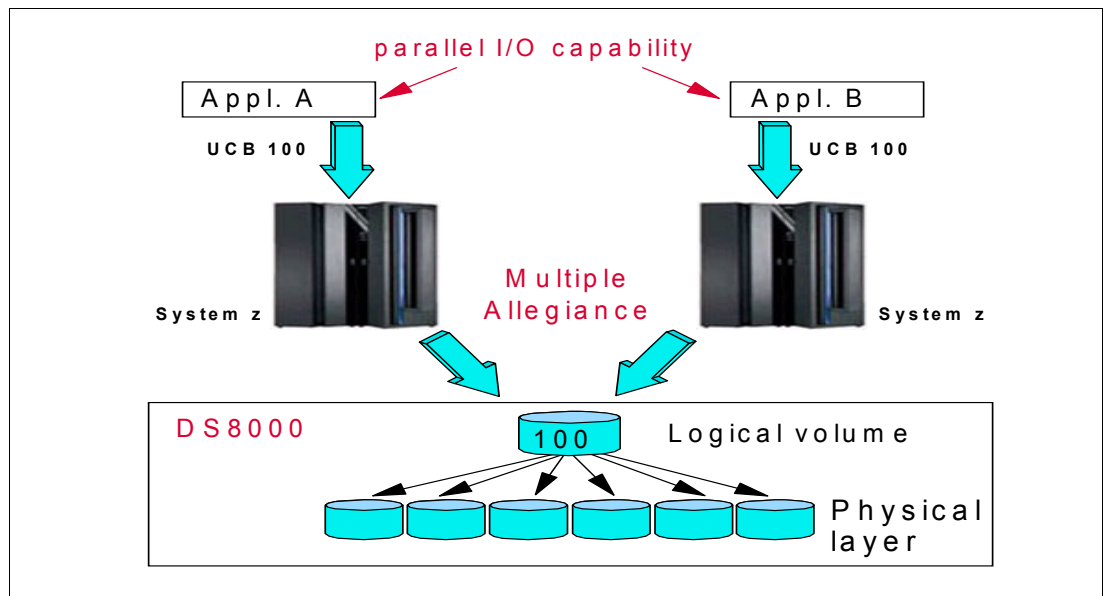


Figure 2-6 Parallel I/O capability with multiple allegiance

The DS8000 accepts multiple I/O requests from separate hosts to the same device address, increasing parallelism and reducing channel overhead. In older storage disk subsystems, a device had an implicit allegiance, that is, a relationship created in the control unit between the device and a channel path group when an I/O operation is accepted by the device. The allegiance causes the control unit to guarantee access (no busy status presented) to the device for the remainder of the channel program over the set of paths associated with the allegiance.

With multiple allegiance, the requests are accepted by the DS8000 and all requests are processed in parallel, unless there is a conflict when writing data to a particular extent of the count key data (CKD) logical volume.

Still, good application software access patterns can improve the global parallelism by avoiding reserves, limiting the extent scope to a minimum, and setting an appropriate file mask, for example, if no write operation is intended.

In systems without multiple allegiance, all except the first I/O request to a shared volume are rejected, and the I/Os are queued in the System z channel subsystem, showing up as PEND time in the RMF reports.

Multiple allegiance provides significant benefits for environments running a Sysplex, or System z systems sharing access to data volumes. Multiple allegiance and PAV can operate together to handle multiple requests from multiple hosts.

## 2.4.2 Parallel access volume (PAV)

PAV is one of the features that was originally introduced with the IBM TotalStorage Enterprise Storage Server (ESS) and that the DS8000 series has inherited. PAV is an optional licensed function of the DS8000 for the z/OS and z/VM operating systems, helping the System z servers that are running applications to concurrently share the same logical volumes.

Two concurrent I/Os to the same volume may serialize against each other, resulting in queuing somewhere within the I/O subsystem. Sometimes the queuing results from some physical resource, such as the channels or disks, being over utilized. At other times it is simply a logical resource, such as the UCBs in the z/OS operating system. A UCB can be used by one and only one I/O at a time. If no UCBs are available, the I/O Supervisor queues the request. PAVs were developed to help eliminate UCB queuing, known as IOSQ time. In the past, control units were not able to service two I/Os to the same volume from two separate systems. Multiple allegiance was developed to deal with volume-level serialization when two systems access the same volume.

In addition, control units must serialize two I/O requests to the same extent range to help avoid data integrity problems. This type of serialization is called *extent range serialization*. In the past the extent range of an I/O was the entire data set extent, but more recently a technique called *shrink wrapping* was developed to minimize the serialization. Shrink wrapping is implemented by the media manager component of DFSMS. For example, if an I/O writes one record on cylinder 10, track 4, and one record on cylinder 15, track 8, media manager will shrink the extent range to wrap around these two track addresses. Concurrent I/Os within this extent range are prohibited, and I/Os outside of this range are allowed. All users of media manager benefit from shrink wrapping, but DB2 goes one step further (if the page size is less than 32 KB) by requesting the system to *bypass serialization*, that is, to bypass extent range serialization. In the example, only two I/Os can read or write cylinder 10, track 4 at precisely the same time, but other tracks are unaffected.

Consider another typical DB2 example. Suppose that one I/O writes 128 4-KB pages that span tracks 5 - 7, while a concurrent I/O writes 128 4-KB pages that span tracks 7 - 9. These two I/Os have an extent conflict. However, because DB2 uses bypass serialization, these two I/Os will not conflict with each other unless they both try to write track 7 at exactly the same time. At present time, DB2 does not use bypass serialization for 32 KB pages because a 32 KB page might span tracks, and DB2 cannot tolerate two separate systems accessing separate parts of the page at the same time.

Now, back to the subject of PAV. Each UCB has an address and each volume has a base UCB, but each I/O may use a pool of UCBs associated with the *logical control unit (LCU)* in z/OS. The maximum number of UCBs for an LCU is 256. Thus, if an LCU has 64 volumes, there may be up to 192 *alias* UCBs, otherwise known as PAV UCBs. Prior to HyperPAV, these aliases were *bound* to a specific base UCB address, and all systems using this volume were forced to use the same mapping of aliases to base address. Dynamic PAVs enabled the aliases to be rebound, but such rebinding was expensive because it had to be coordinated across systems.

In recent years, IBM improved the PAV implementation, calling it HyperPAV. With HyperPAV, aliases are not bound. There is still one pool of up to 256 addresses per LCU, but the same alias may be used for one volume in one instant, and upon I/O completion the alias immediately becomes available for reuse. Furthermore, the same alias address may be used for two separate volumes by two separate z/OS systems at exactly the same time. Thus, HyperPAV significantly improves I/O concurrency within a single system, and especially when multiple systems have separate volume access skews. Given a fixed number of UCBs in an LCU, the best way to avoid IOSQ time is to allocate a small number of UCBs as base addresses and a large number of UCBs as alias addresses. Therefore, EAV and HyperPAV

together help to minimize IOSQ time, which means that if volume reserve contention is not a problem, large volumes are better than small volumes for performance. Volume reserves are typically used to serialize modifications to the volume table of contents (VTOC). System programmers can monitor their systems to see if reserve contention is an issue for them.

## Traditional z/OS behavior without PAV

Traditional storage disk subsystems have allowed for only one channel program to be active to a volume at a time, to help ensure that data being accessed by one channel program cannot be altered by the activities of some other channel program.

Figure 2-7 illustrates the traditional z/OS behavior without PAV, where subsequent simultaneous I/Os to volume 100 are queued, while volume 100 is still busy with a preceding I/O.

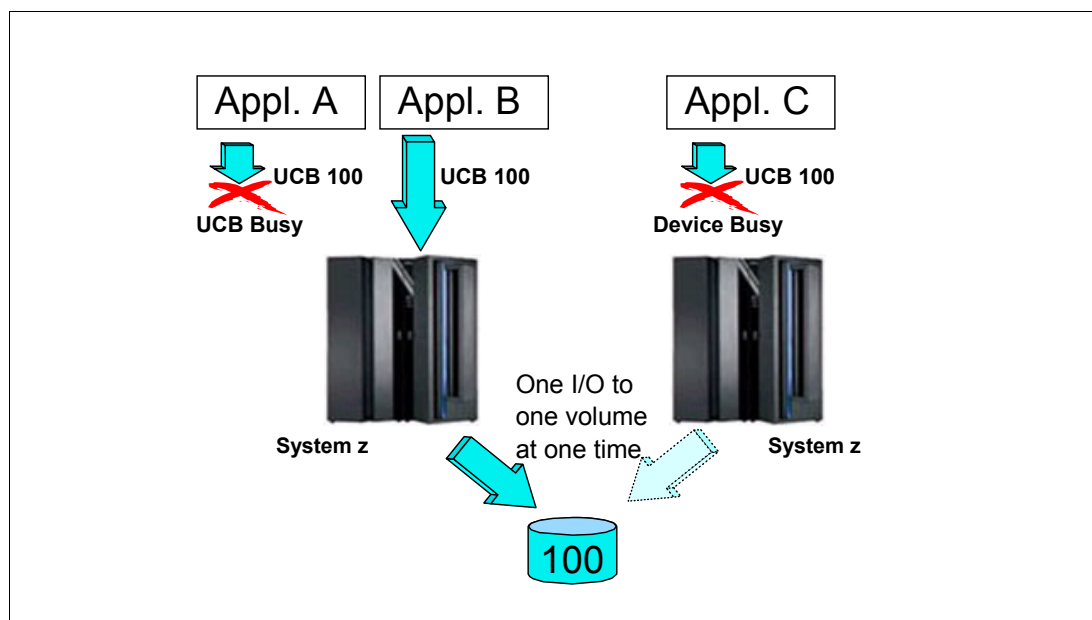


Figure 2-7 Traditional z/OS behavior

From a performance perspective, sending more than one I/O at a time to the storage disk subsystem did not make sense, because the hardware could process only one I/O at a time. Knowing this, the z/OS systems did not try to issue another I/O to a volume, in z/OS represented by a UCB) while an I/O was already active for that volume, as indicated by a UCB busy flag; see Figure 2-7. Not only were the z/OS systems limited to processing only one I/O at a time, but also the storage subsystems accepted only one I/O at a time from separate system images to a shared volume, for the same reasons mentioned previously.

## Parallel I/O capability z/OS behavior with PAV

The DS8000 has the capability to do more than one I/O to a count-key-data (CKD)<sup>2</sup> volume. Using the alias address in addition to the conventional base address, a z/OS host can use several UCBs for the same logical volume instead of one UCB per logical volume. From a DB2 perspective, applications A, B, and C can be replaced with threads 1, 2, and 3, or even a mix such as thread 1, thread 2, and application 1 (Apple.1). The point is that there are

<sup>2</sup> In CKD disk data architecture, the data field stores the user data. Because data records can be variable in length, in CKD they all have an associated count field that indicates the user data record size. The key field enables a hardware search on a key. The commands used in the CKD architecture for managing the data and the storage devices are called channel command words (CCW). It has been used by IBM since 1964 for DASDs on large systems. It contrasts with IBM fixed-block-architecture (FBA) used on disk drives.



multiple requests for access for this volume. For example, base address 100 might have alias addresses 1FF and 1FE, which allow for three parallel I/O operations to the same volume. See Figure 2-8.

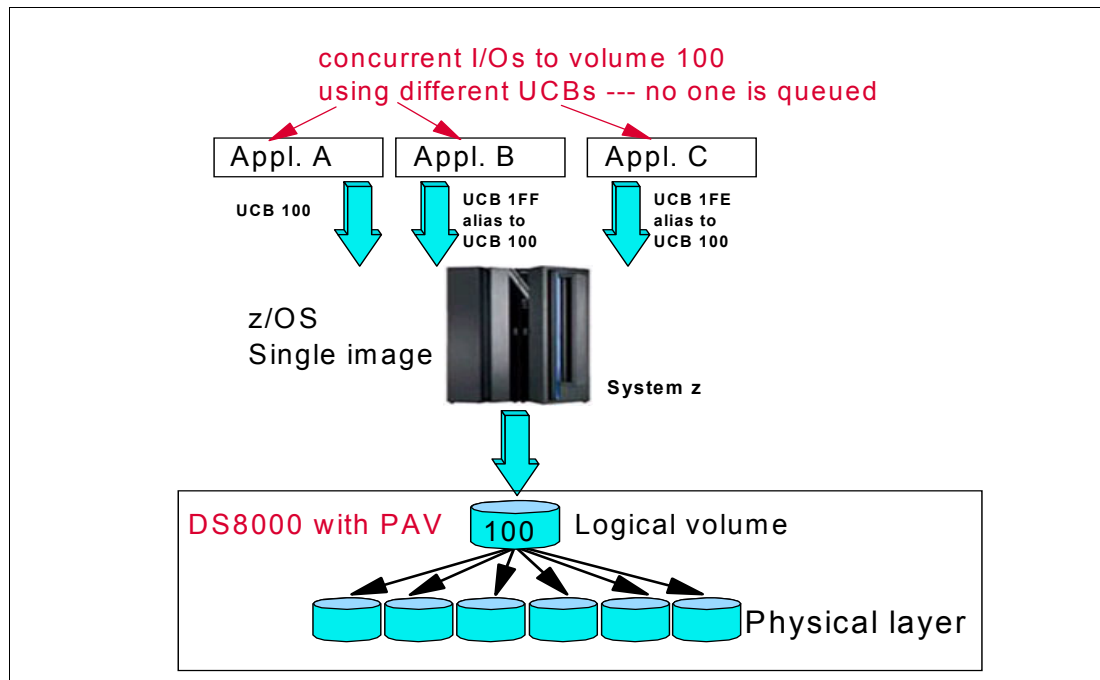


Figure 2-8 z/OS behavior with PAV

This feature that allows parallel I/Os to a volume from one host is called *parallel access volume (PAV)*.

Two concepts are basic in PAV functionality:

- ▶ **Base address:** The base device address is the conventional unit address of a logical volume. Only one base address is associated with any volume.
- ▶ **Alias address:** An alias device address is mapped to a base address. I/O operations to an alias run against the associated base address storage space. No physical space is associated with an alias address. You can define more than one alias per base.

Alias addresses must be defined to the DS8000 and to the I/O definition file (IODF). This association is predefined, and you can add new aliases non disruptively. However, the association between base and alias is not fixed; the alias address can be assigned to a separate base address by the z/OS Workload Manager; for example, with volumes attached to multiple logical partitions (LPARs), every LPAR is required to use the same number of PAVs for each volume. Some system overhead is present when dynamic PAV is moving the aliases for handling dramatically different workloads across the multiple LPARs.

### 2.4.3 z/OS Workload Manager: dynamic PAV tuning

Predicting which volumes should have an alias address assigned, and how many is not always easy. Your software can automatically manage the aliases according to your goals. z/OS can exploit automatic PAV tuning if you are using the z/OS Workload Manager (WLM) in Goal mode. The WLM can dynamically tune the assignment of alias addresses. The Workload Manager monitors the device performance and is able to dynamically reassign alias addresses from one base to another if predefined goals for a workload are not met.

z/OS recognizes the aliases that are initially assigned to a base during the Nucleus Initialization Program (NIP) phase. If dynamic PAVs are enabled, the WLM can reassign an alias to another base by instructing the IOS to do so when necessary. See Figure 2-9.

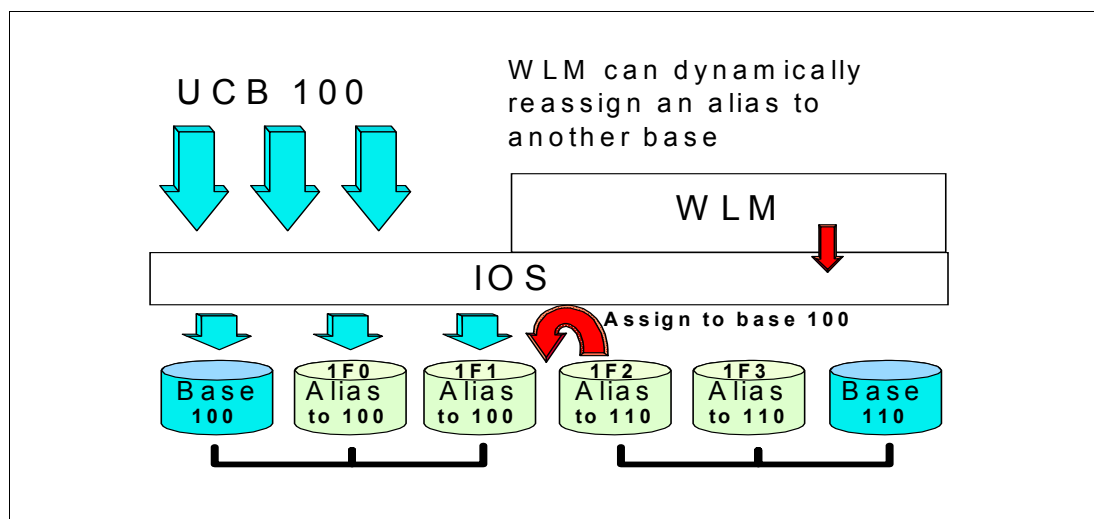


Figure 2-9 WLM assignment of alias addresses

z/OS Workload Manager in Goal mode tracks the system workload and checks if the workloads are meeting their goals established by the installation. See Figure 2-10.

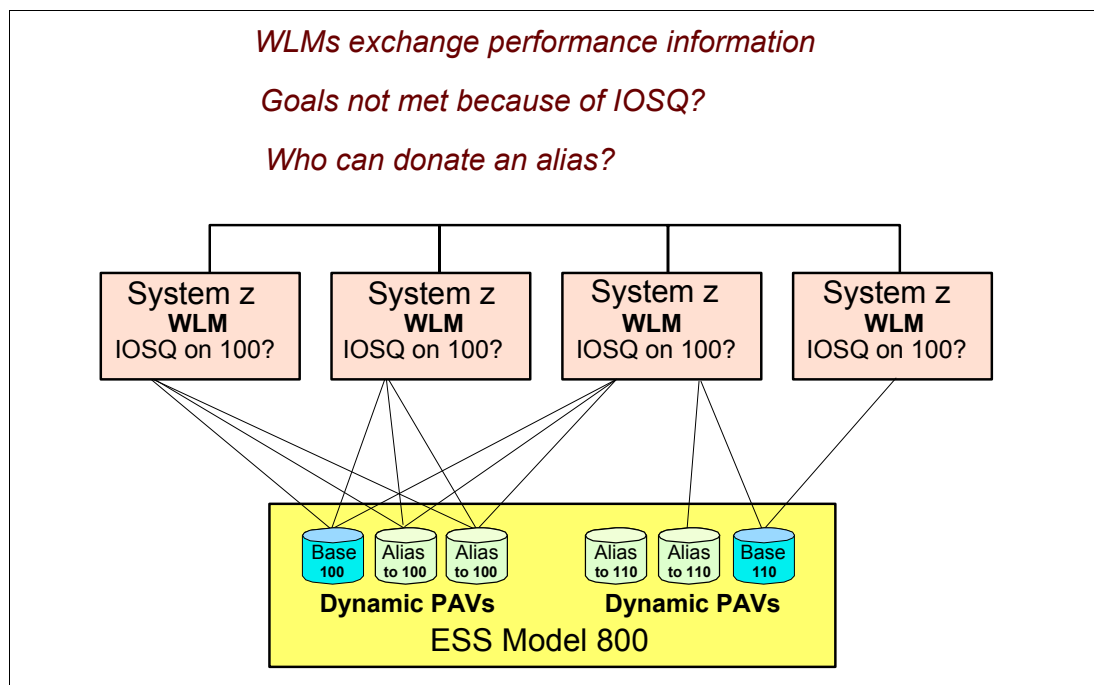


Figure 2-10 Dynamic PAVs in a sysplex

WLM also keeps track of the devices utilized by the various workloads, accumulates this information over time, and broadcasts it to the other systems in the same Sysplex. If WLM determines that any workload is not meeting its goal because of IOS queue (IOSQ) time, WLM attempts to find an alias device that can be reallocated to help this workload achieve its goal. See Figure 2-10.

Two mechanisms can tune the alias assignment:

- ▶ The first is goal-based. This logic attempts to give additional aliases to a PAV-enabled device that is experiencing I/O supervisor (IOS) queue delays and is affecting a service class period that is missing its goal.
- ▶ The second is to move aliases to high-contention PAV-enabled devices from low contention PAV devices. High-contention devices are identified by having a significant amount of IOSQ time. This tuning is based on efficiency rather than directly helping a workload to meet its goal.

As mentioned previously, the WLM must be in Goal mode to cause PAVs to be shifted from one logical device to another.

The movement of an alias from one base to another is serialized within the sysplex. IOS tracks a token for each PAV-enabled device. This token is updated each time an alias change is made for a device. IOS and WLM exchange the token information. When the WLM instructs IOS to move an alias, WLM also presents the token. When IOS has started a move and updated the token, all affected systems are notified of the change through an interrupt.

### **Dynamic PAV on EAV volumes**

Adding more PAV aliases to help access larger EAV volumes contradicts what we want to accomplish with EAV (3390A). The reason for an EAV is simply to increase the amount of addressable storage within an LCU on z/OS. During separate test cases with dynamic PAV on EAV with comparable workloads, we have seen a maximum of 14 alias devices assigned to a base device. Dynamic PAV should be considered when using large volumes (EAV), because it can help PAV management across those volumes, reducing the need to create more PAVs to address performance on EAV volumes. See 2.4.4, “HyperPAV” on page 29 for the best solution for PAV with large volumes (EAV).

### **RMF reporting on PAV**

RMF reports the number of exposures for each device in its Monitor/DASD Activity report and in its Monitor II and Monitor III Device reports. RMF also reports which devices had a change in the number of exposures. RMF reports all I/O activity against the base address, not by the base and associated aliases. The performance information for the base includes all base and alias activity.

## **2.4.4 HyperPAV**

The DS8000 series offers enhancements to parallel access volumes (PAV) with support for HyperPAV, which enables applications to achieve equal or better performance than PAV alone, while also using the same or fewer operating system resources.

With the IBM System Storage DS8000 Turbo model and the IBM server synergy feature, and the HyperPAV together with PAV, multiple allegiance can dramatically improve performance and efficiency for System z environments.

With HyperPAV technology, note the following information about z/OS:

- ▶ z/OS uses a pool of UCB aliases.
- ▶ As each application I/O is requested, if the base volume is busy with another I/O:
  - z/OS selects a free alias from the pool, quickly binds the alias device to the base device, and starts the I/O.
  - When the I/O completes, the alias device is used for another I/O on the LSS or is returned to the free alias pool.

If too many I/Os are started simultaneously, the following events occur:

- ▶ z/OS queues the I/Os at the LCU level<sup>3</sup>.
- ▶ When an exposure frees up that can be used for queued I/Os, they are started.
- ▶ Queued I/O is done within assigned I/O priority.

For each z/OS image within the sysplex, aliases are used independently. WLM is not involved in alias movement so it does not need to collect information to manage HyperPAV aliases.

## PAV and multiple allegiance

Figure 2-11 illustrates the basic characteristics of how traditional PAV operates.

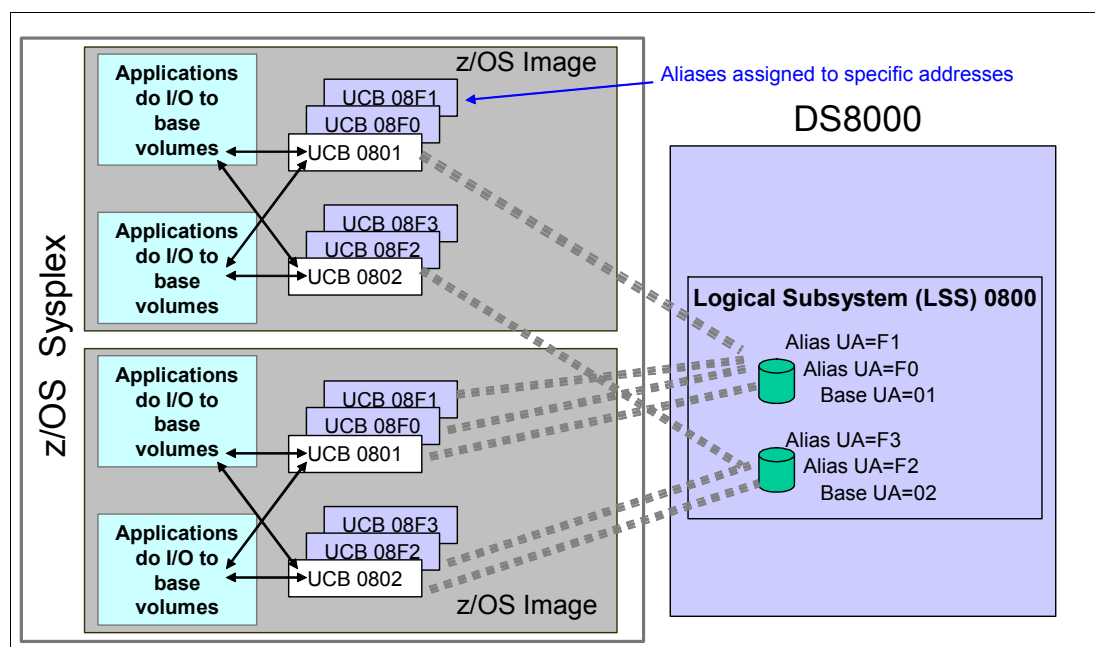


Figure 2-11 Parallel access volumes: basic operational characteristics

Multiple allegiance and PAV allow multiple I/Os to be executed concurrently against the same volume:

- ▶ With multiple allegiance, the I/Os are coming from separate system images.
- ▶ With PAV, the I/Os are coming from the same system image:
  - Static PAV: Aliases are always associated with the same base addresses.
  - Dynamic PAV: Aliases are assigned up front, but can be reassigned to any base address as need dictates by means of the Dynamic Alias Assignment function of the Workload Manager - reactive alias assignment.

With HyperPAV, an on-demand proactive assignment of aliases is possible (Figure 2-12 on page 31).

Previously, dynamic PAV required the WLM to monitor the workload and goals. The WLM detected an I/O bottleneck after some amount of time, and then, the WLM had to coordinate the reassignment of alias addresses within the sysplex and the DS8000. All of this took time, and if the workload was fluctuating or had a burst character, the job that caused the overload

<sup>3</sup> The control units “think” in terms of LSS, but z/OS “thinks” in terms of LCUs. In 6.3, “RMF monitoring” on page 325, the Device Activity Report and I/O Queuing Report in RMF refer to LCUs; control unit cache reports refer to LSS.

of one volume could have ended before the WLM had reacted. In these cases, the IOSQ time was not eliminated completely.

With HyperPAV, the WLM is no longer involved in managing alias addresses. For each I/O, an alias address can be picked from a pool of alias addresses within the same LCU.

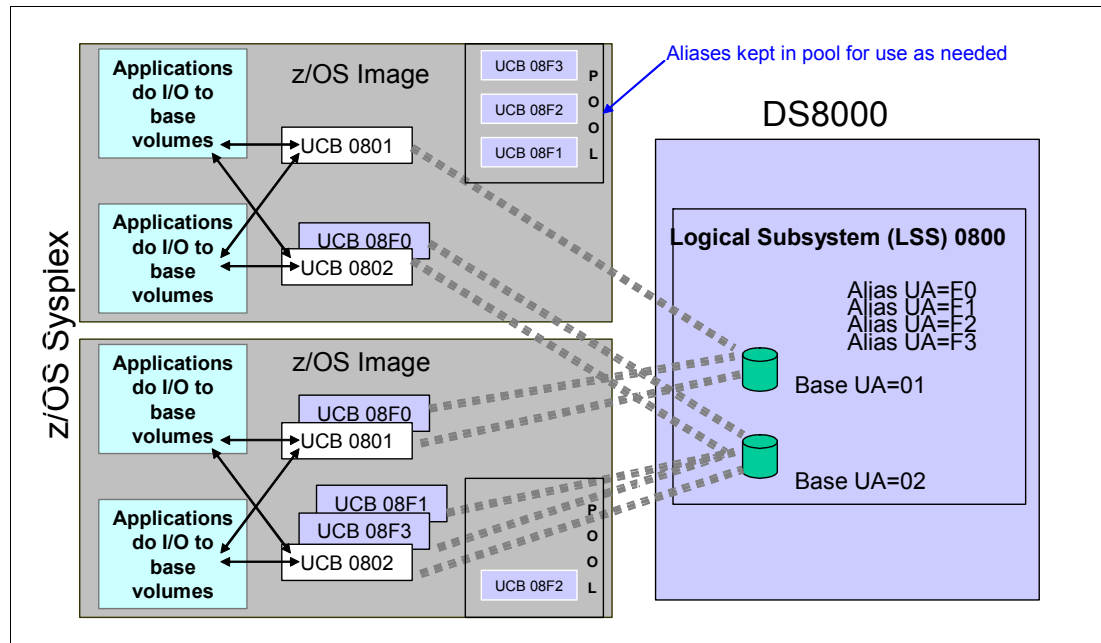


Figure 2-12 HyperPAV: basic operational characteristics

This capability allows separate HyperPAV hosts to use one alias to access separate bases, which reduces the number of alias addresses required to support a set of bases in a System z environment with no latency in targeting an alias to a base. This functionality can also enable applications to achieve better performance than is possible with the original PAV feature alone, while also using the same or fewer operating system resources.

## Benefits of HyperPAV

HyperPAV offers the following benefits:

- ▶ Provides an even more efficient parallel access volumes (PAV) function.
- ▶ Helps clients, who implement larger volumes, to scale I/O rates without the need for additional PAV alias definitions.
- ▶ Uses FICON architecture to reduce overhead, improve addressing efficiencies, and provide storage capacity and performance improvements:
  - More dynamic assignment of PAV aliases improves efficiency.
  - Number of PAV aliases needed might be reduced, taking fewer from the 64 K device limitation and leaving more storage for capacity use.
- ▶ Enables a more dynamic response to changing workloads.
- ▶ Simplifies management of aliases.
- ▶ Enables users to stave off migration to larger volume sizes.

## HyperPAV alias consideration on EAV

HyperPAV provides a far more agile alias management algorithm because aliases are dynamically bound to a base for the duration of the I/O for the z/OS image that issued the I/O.

When an I/O completes, the alias is returned to the pool for the LCU. It then becomes available to subsequent I/Os.

A general suggestion is to start with four aliases for each disk. This amount allows enough accesses to keep the disks 80% busy. The RMF I/O Queuing Report then shows the maximum number of HPAVs used, and if there was an IOSQ time, it shows the maximum depth of the queues. The report identifies whether the aliases are under-configured or over-configured.

Depending on the kind of workload, there is a huge reduction in PAV alias UCBs with HyperPAV. The combination of HyperPAV and EAV allows you to significantly reduce the constraint on the 64K-device<sup>4</sup> address limit and in turn increase the amount of addressable storage available on z/OS. In conjunction with Multiple Subchannel Sets (MSS) on z9® and z10™, you have even more flexibility in device configuration. Consider that the EAV volumes are supported only on z/OS v1.10 and later.

**Note:** For more details about MSS, see *Multiple Subchannel Sets: An Implementation View*, REDP-4387.

## 2.4.5 I/O priority queuing

The concurrent I/O capability of the DS8000 allows it to execute multiple channel programs concurrently, if the data that is accessed by one channel program is not altered by another channel program.

Use I/O priority queuing with dynamic PAV.

### Queuing of channel programs

When the channel programs conflict with each other and must be serialized to ensure data consistency, the DS8000 internally queues channel programs.

This subsystem I/O queuing capability provides significant benefits:

- ▶ Compared to the traditional approach of responding with a *device busy* status to an attempt to start a second I/O operation to a device, I/O queuing in the storage disk subsystem eliminates the overhead associated with posting status indicators and redriving the queued channel programs.
- ▶ Contention in a shared environment is eliminated. Channel programs that cannot execute in parallel are processed in the order that they are queued. A fast system cannot monopolize access to a volume also accessed from a slower system. Each system gets a fair share.

### Priority queuing

I/Os from separate z/OS system images can be queued in a priority order. It is the z/OS WLM that makes use of this priority to privilege I/Os from one system against the others. You can activate I/O priority queuing in WLM Service Definition settings. WLM must run in Goal mode.

When a channel program with a higher priority comes in and is put in front of the queue of channel programs with lower priority, the priority of the low-priority programs is also increased; see Figure 2-13. This technique prevents high-priority channel programs from dominating lower priority ones and gives each system a fair share.

<sup>4</sup> Regarding DASD, 64K is the abbreviation for 64000.

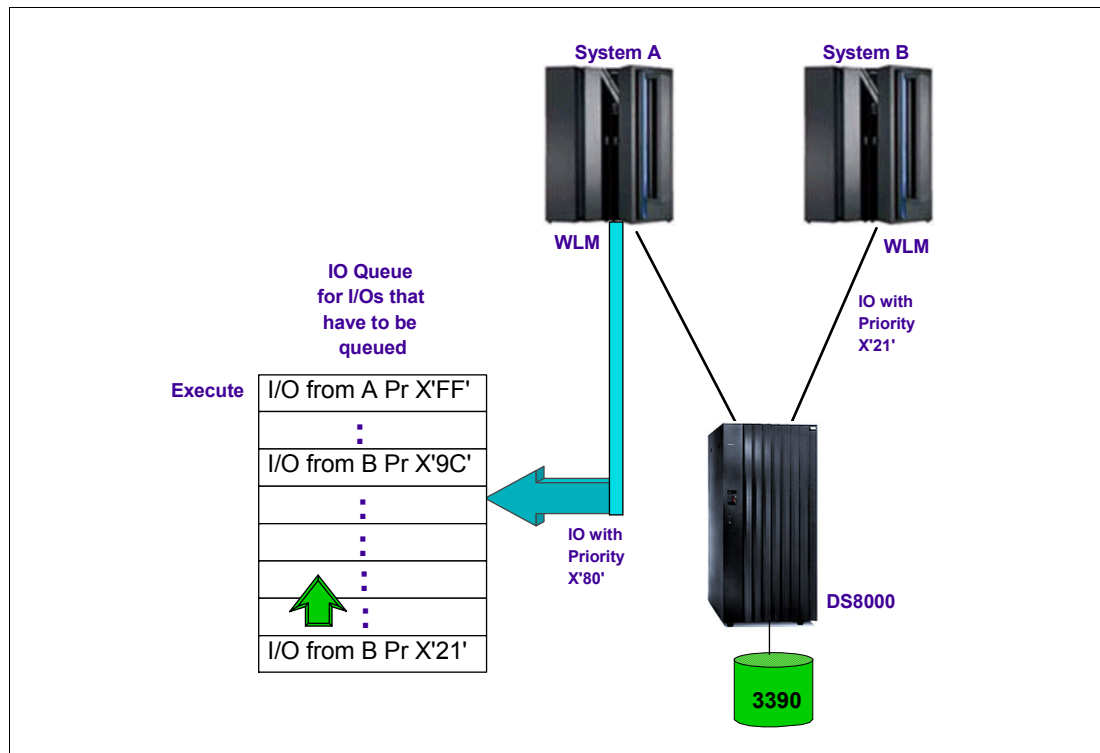


Figure 2-13 I/O priority queuing

## 2.4.6 Dynamic volume expansion

The IBM System Storage DS8000 series supports dynamic volume expansion (DVE), which increases the capacity of existing system volumes, while the volume remains connected to a host system. This capability simplifies data growth by allowing volume expansion without taking volumes offline. Using DVE significantly reduces the complexity of migrating to larger volumes (beyond 65,520 cylinders).

**Note:** For the dynamic volume expansion function, volumes cannot be in copy services relationships (point-in-time copy, FlashCopy SE, Metro Mirror, Global Mirror, Metro/Global Mirror, and z/OS Global Mirror) during expansion.

Use either of the following methods to dynamically grow a volume:

- ▶ The command-line interface (DS CLI)
- ▶ A web browser GUI

**Note:** All systems must be at or above the z/OS V1R10 level for the DVE feature to be used when the systems are sharing the Release 4.0 Licensed Internal Microcode updated DS8000 at an LCU level. Current DS8000 Licensed Internal Microcode is Version 5.4.1.1043.

### Using dynamic volume expansion

Growing an existing volume with the DS8000 with dynamic volume expansion (DVE) is possible. Previously, it was necessary to use migration utilities that required an additional volume for each volume being expanded and that required the data to be moved.

A logical volume can be increased in size while the volume remains online to host systems for the following types of volumes:

- ▶ 3390 model 3 to 3390 model 9
- ▶ 3390 model 9 to EAV volume sizes using z/OS V1R10

Dynamic volume expansion can be used to expand volumes beyond 65,520 cylinders without moving data or causing an application outage.

Dynamic volume expansion is performed by the DS8000 Storage Manager and can be requested using its web GUI. The 3390 volumes may be increased in size, for example from a 3390 model 3 to a model 9 or a model 9 to a model A (EAV). z/OS V1R11 introduces an interface that can be used to make requests for dynamic volume expansion of a 3390 volume on a DS8000 from the system.

**Note:** For the dynamic volume expansion function, volumes cannot be in copy services relationships (point-in-time copy, FlashCopy SE, Metro Mirror, Global Mirror, Metro/Global Mirror, and z/OS Global Mirror) during expansion.

## 2.4.7 Storage management hardware

The use of DFSMS requires storage management hardware that includes direct access storage device (DASD). This section has an overview of storage device categories and a brief introduction to RAID technology.

For many years, DASDs have been the most used storage devices on IBM System z systems and their predecessors, delivering the fast random access to data and high availability that customers have come to expect.

We cover the following types of DASDs:

- ▶ 3380 and 3390
- ▶ Enterprise Storage Server (ESS)
- ▶ DS8000

**Note:** We do not explain the ESS in more detail because effective 28 April 2006, IBM withdrew from marketing the following products:

- ▶ IBM TotalStorage Enterprise Storage Server (ESS) Models 750 and 800
- ▶ IBM Standby Capacity on Demand for ESS offering

The era of tapes began before DASD was introduced. During that time, tapes were used as the primary application storage medium. Today, customers use tapes for such purposes as backup, archiving, or transferring data between companies.

The following types of tape devices are available:

- ▶ IBM TS1130 Tape Drives
- ▶ IBM Tape Library TS3500)
- ▶ IBM Virtualization Engine TS7720 and TS7740

For more information, see *IBM System Storage Solutions Handbook*, SG24-5250-07.

### 3990 volume geometry

z/OS divides the disk space into volumes. Each volume has a volume table of contents (VTOC). The VTOC is a directory that shows where data sets reside and where free space is



on the volume. Before storage virtualization was available, a 3390 volume consisted of one physical spinning disk, and each 3390 volume consisted of 15 tracks per cylinder. This one-to-one mapping is no longer used, but z/OS and the CKD control units that z/OS uses emulate the 3390 track geometry. Disk addresses are logically addressed by cylinder number and track number. Serialization is still done at the track level, and space is still allocated in units of tracks or cylinders.

### **DASD based on RAID technology**

With the introduction of the RAMAC Array in 1994, IBM first introduced storage subsystems for S/390 systems based on RAID technology.

The more modern IBM DASD products, such as ESS, DS6000™, DS8000, and DASD from other vendors, emulate IBM 3380 and 3390 volumes in geometry, capacity of tracks, and number of tracks per cylinder. This emulation makes all other entities “think” they are dealing with real 3380s or 3390s. Among these entities, we have data processing people not working directly with storage, job control language (JCL), MVS commands, open routines, access methods, IOS, and channels. One benefit of this emulation is that it allows DASD manufacturers to implement changes in the real disks, including the geometry of tracks and cylinders, without affecting the way those components interface with DASD. From an operating system point of view, device types are always defined in 3390 compatibility, often with much higher numbers of cylinders, but as 3390s.

We discuss the various RAID implementations. See 2.5, “Redundant Array of Independent Disks (RAID)” on page 35 for more detail of how the RAID technology works.

### **ESS technology**

The IBM TotalStorage Enterprise Storage Server (ESS) is the IBM disk storage server, developed using IBM Seascape architecture. The ESS provides functionality to the family of e-business servers, and also to (that is, Intel®-based and UNIX®-based) families of servers that are not IBM. Throughout these environments, the ESS features unique capabilities that allow it to meet the most demanding requirements of performance, capacity, and data availability that the computing business requires.

### **Seascape architecture**

Seascape architecture is the key to the development of the IBM storage products. Seascape allows IBM to take the best of the technologies developed by the many IBM laboratories and integrate them, thereby producing flexible and upgradable storage solutions. This Seascape architecture design has allowed the IBM TotalStorage Enterprise Storage Server to evolve from the initial E models to the succeeding F models, and to the later 800 models, each featuring new, more powerful hardware and functional enhancements, and always integrated under the same successful architecture with which the ESS was originally conceived. See 2.7, “Seascape architecture” on page 43 for more information.

**Note:** In this publication, we use the terms disk or head disk assembly (HDA) for the real devices, and the terms DASD volumes or DASD devices for the logical 3380/3390s.

## **2.5 Redundant Array of Independent Disks (RAID)**

RAID is a direct access storage architecture where data is recorded across multiple physical disks with redundancy or parity separately recorded, so that no loss of access to data results from the loss of any one disk in the array.

RAID breaks the one-to-one association of volumes with devices. A logical volume is now the addressable entity presented by the controller to the attached systems. The RAID unit maps the logical volume across multiple physical devices. Similarly, blocks of storage on a single physical device may be associated with multiple logical volumes. Because a logical volume is mapped by the RAID unit across multiple physical devices, it is now possible to overlap processing for multiple cache misses to the same logical volume because cache misses can be satisfied by separate physical devices.

The RAID concept involves many small computer system interface (SCSI) disks replacing a big one. The major RAID advantages are as follows:

- ▶ Performance (because of parallelism)
- ▶ Cost (SCSI are commodities)
- ▶ System z compatibility
- ▶ Environment (space and energy)

However, the use of RAID increased the chances of malfunction because of media and disk failures and because the logical device now resided on many physical disks. The solution was redundancy, which wasted space and caused performance problems as *write penalty* and *free space reclamation*. To address this performance issue, large caches were implemented.

**Note:** The ESS storage controllers use the RAID architecture that enables multiple logical volumes to be mapped on a single physical RAID group. If required, you can still separate data sets on a physical controller boundary for the purpose of availability.

Except for RAID 1, each manufacturer sets the number of disks in an array. An *array* is a set of logically related disks, where a parity applies.

Various implementations are certified by the RAID Architecture Board:

- RAID 1** This type has simple disk mirroring, such as dual copy.
- RAID 3** This type has an array with one dedicated parity disk and only one I/O request at a time, with intra-record striping, which means that the written physical block is striped and each piece (together with the parity) is written in parallel in each disk of the array. The access arms move together. It has a high data rate and a low I/O rate.
- RAID 5** This type has an array with one distributed parity (there is no dedicated disk for parities). It does I/O requests in parallel with extra-record striping, meaning each physical block is written in each disk. The access arms move independently. It has strong caching to avoid write penalties; that is, four disk I/Os per write. RAID 5 has a high I/O rate and a medium data rate. RAID 5 is used by the IBM DS8000 with 8-disk arrays in the majority of configurations.

RAID 5 performs the following tasks:

- ▶ It reads data from an undamaged disk. This is one single disk I/O operation.
- ▶ It reads data from a damaged disk, which implies (n-1) disk I/Os, to recreate the lost data where n is the number of disks in the array.
- ▶ For every write to an undamaged disk, RAID 5 does four I/O operations to store a correct parity block; this is called a write penalty. This penalty can be relieved with strong caching and a slice triggered algorithm (coalescing disks updates from cache into a single parallel I/O).
- ▶ For every write to a damaged disk, RAID 5 does n-1 reads and one parity write.

Figure 2-14 summarizes the characteristics of RAID 1, RAID 3, and RAID 5.

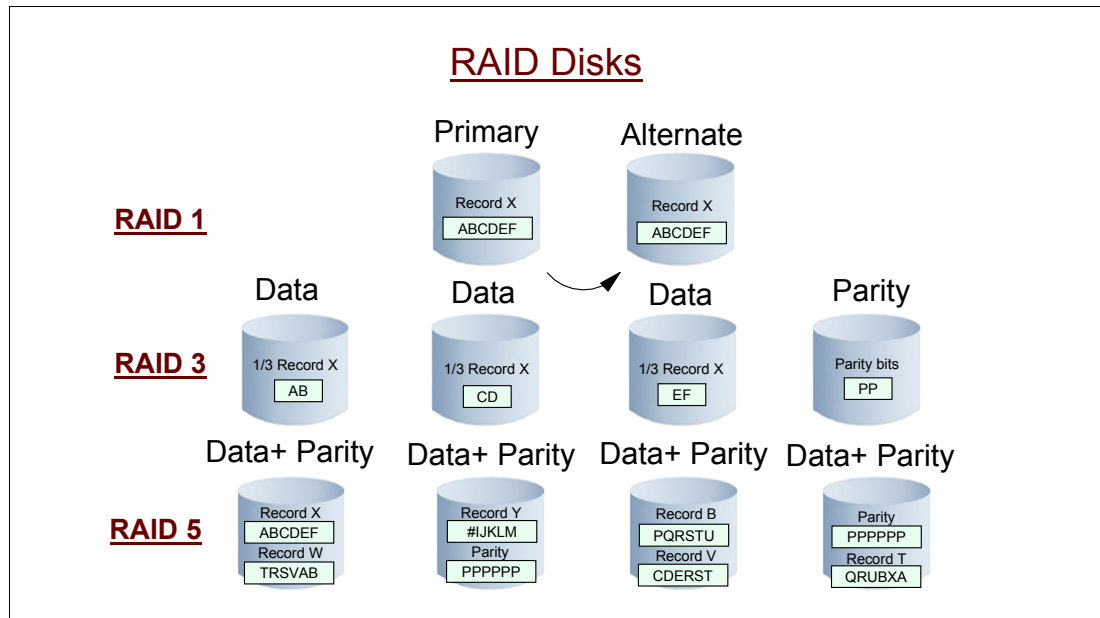


Figure 2-14 RAID

- RAID 6** This type has an array with two distributed parity and I/O requests in parallel with extra-record striping. Its access arms move independently (Reed-Solomon P-Q parity). The write penalty is greater than RAID 5, with six I/Os per write.
- RAID 6+** This type is without write penalty (because of log-structured file, or LFS), and has background free-space reclamation. The access arms all move together for write operations. It is used by the RVA controller.
- RAID 10** This type has a new RAID architecture designed to give performance for striping and has redundancy for mirroring. RAID 10 is implemented in the DS8000 series.

The various RAID types have different performance, economy, and availability characteristics. For instance, RAID 1 uses 50% of disk capacity for data protection, and gets the best performance for both reads and writes. RAID 3 (three data disks and one parity) uses only 25% of capacity for protection. RAID 5 requires four I/Os for every access. RAID 6 with double parity requires six I/Os for every write and therefore does not perform well in a high-write environment.

**Note:** Data striping (stripe sequential physical blocks in separate disks) is sometimes called RAID 0, but it is not a real RAID because there is no redundancy, that is, no parity bits.

## 2.5.1 RAID 6 overview

RAID 6 protection provides more fault tolerance than RAID 5 in the case of disk failures and uses less raw disk capacity than RAID 10.

RAID 6 allows for additional fault tolerance by using a second independent distributed parity scheme (dual parity). Data is striped on a block level across a set of drives, similar to RAID 5 configurations, and a second set of parity is calculated and written across all the drives, as illustrated in Figure 2-15.

### One Stride with 5 data drives (5 + P + Q):

Drives	1	2	3	4	5	P	Q
	0	1	2	3	4	P00	P01
	5	6	7	8	9	P10	P11
	10	11	12	13	14	P20	P21
	15	16	17	18	19	P30	P31
							P41

$P00 = 0+1+2+3+4$ ;  $P10 = 5+6+7+8+9$ ;... (parity on block level across a set of drives)

$P01 = 9+13+17+0$ ;  $P11 = 14+18+1+5$ ;... (parity across all drives)

$P41 = 4+8+12+16$

**NOTE: For illustrative purposes only – implementation details may vary**

Figure 2-15 One RAID 6 stride

RAID 6 is best used in combination with large capacity disk drives, such as the 500 GB Fibre ATA (FATA) or SATA drives, because they have a longer rebuild-time, but it may also be used with FC drives, when the primary concern is to have a higher reliability.

Comparing RAID 6 to RAID 5 performance, gives about the same results on read operations. For random writes, the throughput of a RAID 6 array is only around two thirds of a RAID 5, given the additional parity handling. Workload planning is especially important before implementing RAID 6 for write-intensive applications including copy services targets and FlashCopy SE repositories. Yet, when properly sized for the I/O demand, RAID 6 is a considerable reliability enhancement.

## 2.5.2 RAID 10 overview

RAID 10 is not as commonly used as RAID 5, mainly because more raw disk capacity is needed for every gigabyte of effective capacity.

RAID 10 provides high availability by combining features of RAID 0 and RAID 1. RAID 0 optimizes performance by striping volume data across multiple disk drives at a time. RAID 1 provides disk mirroring, which duplicates data between two disk drives. By combining the features of RAID 0 and RAID 1, RAID 10 provides a second optimization for fault tolerance. Data is striped across half of the disk drives in the RAID 1 array. The same data is also striped across the other half of the array, creating a mirror. Access to data is preserved if one disk in each mirrored pair remains available. RAID 10 offers faster data reads and writes than RAID 5 because it does not need to manage parity. However, with half the DDMs in the group used for data and the other half to mirror that data, RAID 10 disk groups have less capacity than RAID 5 disk groups.

A typical area of operation for RAID 10 is workloads with a high random write ratio.

## 2.5.3 RAID implementation in the DS8000

DS8000 supports RAID 5, 6, and 10.

### RAID 5 implementation in the DS8000

In a DS8000, a RAID 5 array built on one array site will contain either seven or eight disks, depending whether the array site is supplying a spare. A seven-disk array effectively uses one disk for parity, so it is referred to as a 6+P array (where the P stands for parity). The reason only seven disks are available to a 6+P array is that the eighth disk in the array site that is used to build the array was used as a spare. This is referred to as a 6+P+S array site (where the S stands for spare). An 8-disk array also effectively uses one disk for parity, so it is referred to as a 7+P array.

#### *Drive failure*

When a disk drive module fails in a RAID 5 array, the device adapter starts an operation to reconstruct the data that was on the failed drive onto one of the spare drives. The spare that is used will be chosen based on a smart algorithm that looks at the location of the spares and the size and location of the failed DDM. The rebuild is performed by reading the corresponding data and parity in each stripe from the remaining drives in the array, then performing an exclusive-OR operation to recreate the data, and then writing this data to the spare drive.

While this data reconstruction is occurring, the device adapter can still service read and write requests to the array from the hosts. There might be some degradation in performance while the sparing operation is in progress because certain device adapter (DA) and switched network resources are used to do the reconstruction. Because of the switch-based architecture, this effect is minimal. Additionally, any read requests for data on the failed drive require data to be read from the other drives in the array, and then the DA performs an operation to reconstruct the data.

Performance of the RAID 5 array returns to normal when the data reconstruction onto the spare device completes. The time taken for sparing can vary, depending on the size of the failed DDM and the workload on the array, the switched network, and the DA. The use of arrays across loops (AAL) both speeds up rebuild time and decreases the impact of a rebuild.

### RAID 6 implementation in the DS8000

A RAID 6 array in one array site of a DS8000 can be built on either seven or eight disks:

- ▶ In a seven-disk array, two disks are always used for parity; the eighth disk of the array site is needed as a spare. This kind of a RAID 6 array is hereafter referred to as a 5+P+Q+S array (where P and Q stand for parity and S stands for spare).
- ▶ A RAID 6 array, consisting of eight disks is built, when all necessary spare drives are available. An eight-disk RAID 6 array also always uses two disks for parity, so it is referred to as a 6+P+Q array.

#### *Drive failure*

When a disk drive module (DDM) fails in a RAID 6 array, the DA starts to reconstruct the data of the failing drive onto one of the available spare drives. A smart algorithm determines the location of the spare drive to be used, depending on the size and the location of the failed DDM. After the spare drive has replaced a failed one in a redundant array, the recalculation of the entire contents of the new drive is performed by reading the corresponding data and parity in each stripe from the remaining drives in the array and then writing this data to the spare drive.

During the rebuilding of the data on the new drive, the DA can still handle I/O requests of the connected hosts to the affected array. Some performance degradation can occur during the reconstruction because some device adapters and switched network resources are used to do the rebuilding. Because of the switch-based architecture of the DS8000, this effect is minimal. Additionally, any read requests for data on the failed drive require data to be read from the other drives in the array, and then the DA performs an operation to reconstruct the data. Any subsequent failure during the reconstruction within the same array (second drive failure, second coincident medium errors, or a drive failure and a medium error) can be recovered without loss of data.

Performance of the RAID 6 array returns to normal when the data reconstruction on the spare device has completed. The rebuild time varies, depending on the size of the failed DDM and the workload on the array and the DA. The completion time is comparable to a RAID 5 rebuild process, but slower than rebuilding a RAID 10 array in the case of a single drive failure.

Note that RAID 5 can tolerate loss of only one device; RAID 6 allows for loss of two devices at the same time.

## **RAID 10 implementation in the DS8000**

In the DS8000, the RAID 10 implementation is achieved using either six or eight DDMs. If spares exist on the array site, six DDMs are used to make a three-disk RAID 0 array, which is then mirrored. If spares do not exist on the array site, eight DDMs are used to make a four-disk RAID 0 array, which is then mirrored.

### ***Drive failure***

When a disk drive module (DDM) fails in a RAID 10 array, the controller starts an operation to reconstruct the data from the failed drive onto one of the hot-spare drives. The spare that is used is chosen based on a smart algorithm that looks at the location of the spares and the size and location of the failed DDM. Remember that a RAID 10 array is effectively a RAID 0 array that is mirrored. Thus, when a drive fails in one of the RAID 0 arrays, we can rebuild the failed drive by reading the data from the equivalent drive in the other RAID 0 array.

While this data reconstruction is occurring, the DA can still service read and write requests to the array from the hosts. There might be some degradation in performance while the sparing operation is in progress, because some DA and switched network resources are used to do the reconstruction. Because of the switch-based architecture of the DS8000, this effect is minimal. Read requests for data on the failed drive are not be affected because they can all be directed to the good RAID 1 array.

Write operations are not affected. Performance of the RAID 10 array returns to normal when the data reconstruction onto the spare device completes. The amount of time for sparing can vary, depending on the size of the failed DDM and the workload on the array and the DA. In relation to a RAID 5, RAID 10 sparing completion time is a little faster. The reason is because rebuilding a RAID 5 6+P configuration requires six reads plus one parity operation for each write, whereas a RAID 10 3+3 configuration requires one read and one write operation (essentially a direct copy).

### ***Arrays across loops***

The DS8000 implements the concept of arrays across loops (AAL). With AAL, an array site is actually split into two halves. Half of the site is located on the first disk loop of a DA pair and the other half is located on the second disk loop of that DA pair. It is implemented primarily to maximize performance. However, in RAID 10, we are able to take advantage of AAL to provide a higher level of redundancy. The DS8000 RAS code deliberately ensures that one RAID 0 array is maintained on each of the two loops that are created by a DA pair. This approach means that in the extremely unlikely event of a complete loop outage, the DS8000

does not lose access to the RAID 10 array. The reason is that while one RAID 0 array is offline, the other remains available to service disk I/O.

## 2.6 HyperSwap

In the last decade, we have observed the dramatic increase in the amount of storage per disk subsystem. With the average disk subsystem from all vendors containing 45 TB or more of data, the impact of a possible disk subsystem failure becomes more wide. Two approaches can address this problem.

One approach is to build a *higher quality* storage subsystem, which all vendors have and will continue to provide. However, with more intelligence delegated to the storage subsystem, there is more licensed control code and software running on it. This code can be a single point of failure at the entire storage subsystem level.

Another approach is to assume that hardware will eventually fail, and software will eventually work, therefore allow the masking of storage subsystem failures and preventing their impact to critical production applications. This approach means providing a high-availability solution such as the IBM HyperSwap technology with GDPS/PPRC HyperSwap, which was made available to the marketplace in 2002.

The HyperSwap function, shown in Figure 2-16 on page 42, can broaden the continuous availability attributes of z/OS by extending the Parallel Sysplex redundancy to disk subsystems.

Planned HyperSwap function provides capabilities, as follows:

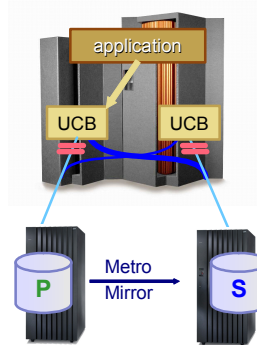
- ▶ Transparently switch all primary Metro Mirror (PPRC) disk subsystems with the secondary Metro Mirror disk subsystems for a planned reconfiguration.
- ▶ Perform disk configuration maintenance and planned site maintenance without requiring any applications to be quiesced.

Unplanned HyperSwap function contains additional functions to transparently switch to use secondary Metro Mirror disk subsystems in the event of unplanned outages of the primary Metro Mirror disk subsystems.

Unplanned HyperSwap support allows production systems to remain active during a disk subsystem failure. Disk subsystem failures no longer constitute a single point of failure for an entire Parallel Sysplex.

Basic HyperSwap in z/OS 1.9 is managed by TPC-R software available with z/OS. Basic HyperSwap can scale to all volumes within the sysplex including all system volumes. It is limited to IBM Disk storage subsystems (ESS 800, DS6000 and DS8000) and is available as a PTF on z/OS 1.9.

## IBM DS8000 Metro Mirror and Basic HyperSwap



- Ability to swap enterprise class System z Disk Subsystems in seconds.
  - HyperSwap substitutes Metro Mirror secondary for primary device
  - No operator interaction, TPC-R Basic Edition managed
  - Designed to scale to multi-thousands of z/OS volumes
  - Includes volumes with SYSRES, page data sets, catalogs
  - Non-disruptive - applications keep using same device addresses
  - HyperSwap integration with z/OS yielding Higher Availability for z/OS.
  - IBM DiskSubsystem (ESS 800, DS6000 & DS8000)
  - z/OS 1.9+ Volumes ONLY.
  - Source & Target Subsystem(s) on Same Data Center Floor

**Higher Availability for System z**

Figure 2-16 Basic HyperSwap

Basic HyperSwap runs on z/OS 1.9 with the Basic HyperSwap SPE installed and uses TPC-R Basic Edition v3.4 (no cost feature of z/OS 1.9). TPC-R Basic Edition also requires a database, either z/OS DB2 V8 or later or the Apache Derby database (free database software), and either the free IBM System Services Runtime environment (SSRE) or z/OS WAS v6.1 or later. TPC-R Basic Edition is fully upgradable to the full versions of TPC-R.

GDPS/PPRC HyperSwap Manager is an entry level solution out of IBM GTS Services, which for a one-time fixed price provides the HyperSwap function for z/OS zVM and zLinux environments. Included is any vendor's storage subsystem that supports the IBM PPRC architecture. This solution is a NetView® application that exploits System Automation for z/OS. This solution is for single and multiple sites and is fully upgradeable to the full GDPS. A fixed-price, scaled-down, single-purpose licensed version of the NetView and System Automation product is available for HyperSwap Manager.

GDPS/PPRC HyperSwap is an end-to-end solution for the highest availability intended for z/OS, zVM, and zLinux environments from the IBM GTS Services team. Planned and unplanned HyperSwap functions, complete parallel sysplex host management, workload management, coordinated network switches, and end-to-end management of simple and complex data replication environments are provided. GDPS/PPRC HyperSwap works in conjunction with GDPS/MGM and GDPS/XRC to provide High Availability locally combined with disaster recovery protection. Single and multi-site workloads are both supported by GDPS/PPRC HyperSwap. Full GDPS/PPRC HyperSwap, such as the GDPS/PPRC HyperSwap Manager solution, also supports all storage vendors hardware that supports the IBM PPRC architecture.



## 2.7 Seascape architecture

The IBM Enterprise Storage Server's *architecture for e-business* design is based on the IBM storage enterprise architecture, Seascape.

The Seascape architecture defines next-generation concepts for storage by integrating modular building block technologies from IBM, including disk, tape, and optical storage media, powerful processors, and rich software. Integrated Seascape solutions are highly reliable, scalable, and versatile, and support specialized applications on servers ranging from PCs to super computers. Virtually all types of servers can concurrently attach to the ESS, including iSeries® and AS/400® systems. As a result, ESS can be the external disk storage system of choice for AS/400 and iSeries systems in heterogeneous SAN environments.

Seascape has three basic concepts:

- ▶ Powerful storage server
- ▶ Snap-in building blocks
- ▶ Universal data access

DFSMS provides device support for the IBM 2105 Enterprise Storage Server (ESS), a high-end storage subsystem. The ESS storage subsystem succeeded the 3880, 3990, and 9340 subsystem families. Designed for mid-range and high-end environments, the ESS gives you large capacity, high performance, continuous availability, and storage expendability.

The ESS was the first of the Seascape architecture storage products to attach directly to IBM System z and open system platforms. The Seascape architecture products come with integrated storage controllers. These integrated storage controllers allow the attachment of physical storage devices that emulate 3390 Models 2, 3, and 9, or provide 3380 track compatibility mode.

### Powerful storage server

The storage system is intelligent and independent, and it can be reached by channels or through the network. It is powered by a set of fast RISC processors.

### Snap-in building blocks

Each Seascape product consists of building blocks, such as the following examples:

- ▶ Scalable n-way RISC server, PCI-based, which provides the logic of the storage server
- ▶ Memory cache from RISC processor memory
- ▶ Channel attachments, such as FC-AL, SCSI, ESCON, FICON and SSA
- ▶ Network attachments, such as Ethernet, FDDI, TR, and ATM.
- ▶ The channel and network attachments can also implement functions, that is, a mix of network interfaces (to be used as a remote and independent storage server) and channel interfaces (to be used as a storage controller interface).
- ▶ Software building blocks, such as an AIX® subset, Java™ applications, and Tivoli® Storage Manager. High level language (HLL) is more flexible than microcode, and is easier to write and maintain.
- ▶ Storage adapters, for mixed storage devices technologies
- ▶ Storage device building blocks, such as serial disk (7133), TS1100 Tape Drive Family, and optical (3995)
- ▶ Tape Automation (TS3500)

## Universal data access

Universal data access allows a wide array of connectivity, such as z/OS, UNIX, Linux®, and OS/400®, to common data. Three types of universal access are available, as follows:

- Storage sharing

Physical storage (DASD or tape) is statically divided into fixed partitions available to a given processor. It is not a software function. The subsystem controller knows which processors own which storage partitions. In a sense, only capacity is shared, not data; one server cannot access the data of the other server. A requirement is that the manual reassignment of storage capacity between partitions be simple and nondisruptive.

The benefits are as follows:

- Higher quantities can be purchased with greater discounts.
- Only one type of storage has to be managed.
- Static shifting of capacity is done as needed.

The drawbacks are as follows:

- Higher price for SCSI data
- Collocation at 20 meters of the SCSI servers
- No priority concept between z/OS and UNIX/NT I/O requests

- Data copy sharing

Data copy sharing is an interim data replication solution (waiting for a true data sharing) done by data replication from one volume accessed by a platform to another volume accessed by another platform. The replication can be done through software or hardware.

The following methods can implement data copy sharing:

- Network: Use network data transfer, such as SNA or TCP/IP. This method has drawbacks, such as CPU and network overhead; it is still slow and expensive for massive data transfer.
- Direct channel: Direct data transfer between the processors involved using channel or bus capabilities, referred to as *bulk data transfer*.
- Shared storage transfer: Writing an intermediate flat file by software into the storage subsystem cache, that is read (and translated) by the receiving processor, so the storage is shared.

- True data sharing

For data sharing between multiple platforms for read/write of a single copy that addresses the complex issues of mixed data types, file structures, databases, and SCPs, there is no available solution.

## 2.8 Architecture of the DS8000 series

The architecture of the DS8000 series is built on the Seascape architecture that connects these components. See Figure 2-17 on page 45.

## DS8000 Architecture

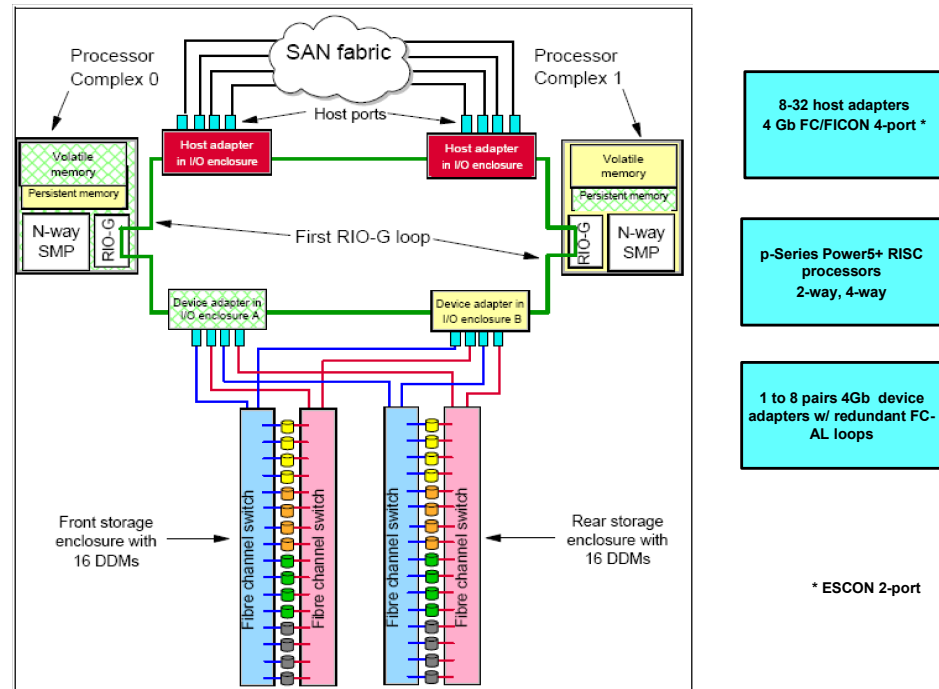


Figure 2-17 DS8000 architecture

To access the disk subsystem, each complex uses several four-port Fibre Channel arbitrated loop (FC-AL) device adapters. A DS8000 can have up to sixteen of these adapters arranged into eight pairs. Each adapter connects the complex to two separate switched Fibre Channel networks. Each switched network attaches disk enclosures that each contain up to 16 disks. Each enclosure contains two 20-port Fibre Channel switches. Of these 20 ports, 16 are used to attach to the 16 disks in the enclosure and the remaining four are used to either interconnect with other enclosures or to the device adapters. Each disk is attached to both switches. Whenever the device adapter connects to a disk, it uses a switched connection to transfer data. This means that all data travels through the shortest possible path.

The attached hosts interact with software which is running on the complexes to access data on logical volumes. Each complex hosts at least one instance of this software (called a *server*), which runs in a logical partition (LPAR). The servers manage all read and write requests to the logical volumes on the disk arrays. During write requests, the servers use fast-write, in which the data is written to volatile memory on one complex and persistent memory on the other complex. The server then reports the write as complete before it has been written to disk. This provides much faster write performance. Persistent memory is also called *nonvolatile storage (NVS)*. However, the servers have not only a volatile memory to deliver fast write there have also a much bigger memory, *cache*, to deliver fast read too.

The DS8000 architecture demonstrates the sophisticated virtualization implemented from IBM into the actual storage subsystem. See 2.9, "Disk virtualization" on page 46 for more detail about the virtualization concept.

## 2.9 Disk virtualization

In a fast changing world, to react quickly to changing business conditions, IT infrastructure must allow for on demand changes. Virtualization is key to an on demand infrastructure. However, with regard to virtualization, many vendors are talking about separate things.

For this chapter, the definition of *virtualization* is the abstraction process from the physical disk drives to a logical volume that is presented to the hosts and servers in a way so they *see it as though it were* a physical disk.

As mentioned at the beginning of this chapter, the storage area moved from a physical mapped infrastructure to a virtual mapped infrastructure. This paradigm change was necessary to get all the new functions (large volume support, Extended Address Volume, RAID, and others) described previously. In this section, we look at the virtualization concepts as they apply to the IBM System Storage DS8000 series.

This section covers the following topics:

- ▶ Storage system virtualization
- ▶ Abstraction layers for disk virtualization
- ▶ Array sites
- ▶ Arrays
- ▶ Ranks
- ▶ Extent pools
- ▶ Logical volumes
- ▶ Allocation, deletion, and modification of CKD volumes
- ▶ Data striping by access method
- ▶ Dynamic volume expansion
- ▶ Logical subsystems
- ▶ Summary of the virtualization hierarchy
- ▶ Benefits of virtualization

### 2.9.1 Storage system virtualization

The DS8000 LPAR Models (9A2 and 9B2) support LPAR mode, with up to two storage logical partitions (storage images) on a physical storage system unit. The storage facility image (SFI), or storage LPAR, is a virtual storage subsystem with its own copy of Licensed Internal Code, which consists of the AIX kernel and the functional code. Both SFIs share the physical hardware, and the LPAR Hypervisor manages this sharing. With Licensed Machine Code 5.4.xx.xx, the resources can be shared in a 50/50 percent ratio or a 25/75 percent ratio.

Just as in non-LPAR mode, where two SMPs are running an AIX kernel and forming a storage complex with two servers, server 0 and server 1, an SFI is a storage complex of its own. Because it does not own the physical hardware (the storage unit), you can think of it as a virtual storage system. Each SFI has a server 0 and a server 1 and is totally separate from the other SFI by the LPAR Hypervisor. Disk drives and arrays are owned by one or the other SFI; they cannot be shared. Figure 2-18 on page 47 illustrates the storage LPAR concept. In 2.9.2, “Abstraction layers for disk virtualization” on page 47, server 0 or server 1 can also mean server 0 or server 1 of a SFI running in an LPAR.

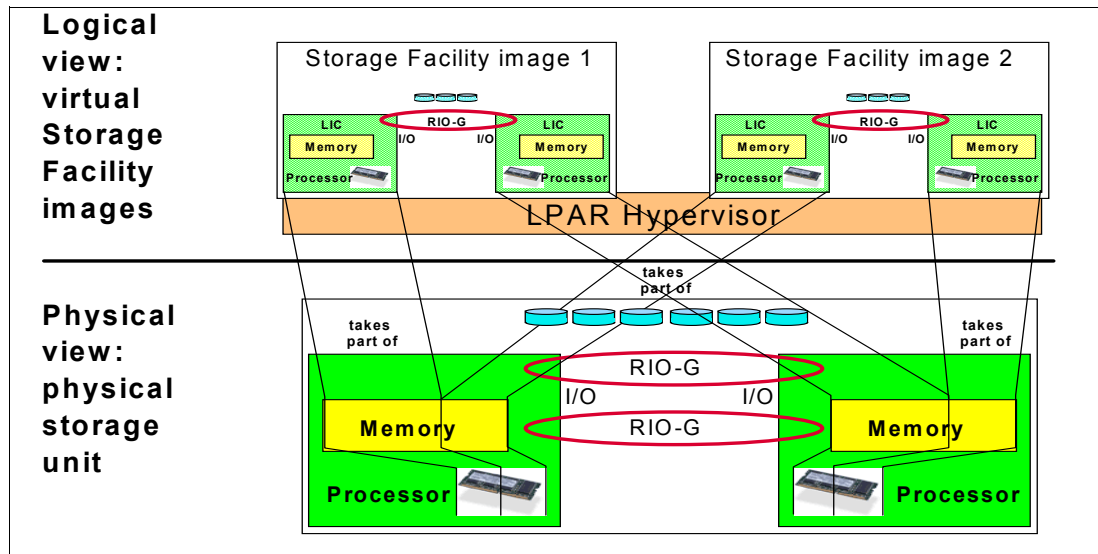


Figure 2-18 Storage facility virtualization

## 2.9.2 Abstraction layers for disk virtualization

In this section, virtualization is about the process of preparing a bunch of physical disk drives (DDMs) to become an entity that can be used by an operating system, which means we are talking about the creation of volumes.

The DS8000 is populated with switched FC-AL disk drives that are mounted in disk enclosures. You order disk drives in groups of 16 drives of the same capacity and RPM. The disk drives can be accessed by a pair of device adapters. Each device adapter has four paths to the disk drives. The four paths provide two FC-AL device interfaces, each with two paths, such that either path can be used to communicate with any disk drive on that device interface (meaning that the paths are redundant). One device interface from each device adapter is connected to a set of FC-AL devices such that either device adapter has access to any disk drive through two independent switched fabrics (meaning that the device adapters and switches are redundant).

Each device adapter has four ports, and because device adapters operate in pairs, there are eight ports or paths to the disk drives. All eight paths can operate concurrently and could access all disk drives on the attached fabric. In normal operation, however, disk drives are typically accessed by one device adapter. Which device adapter owns the disk is defined during the logical configuration process. This avoids any contention between the two device adapters for access to the disks. Figure 2-19 on page 48 shows the physical layer on which virtualization is based.

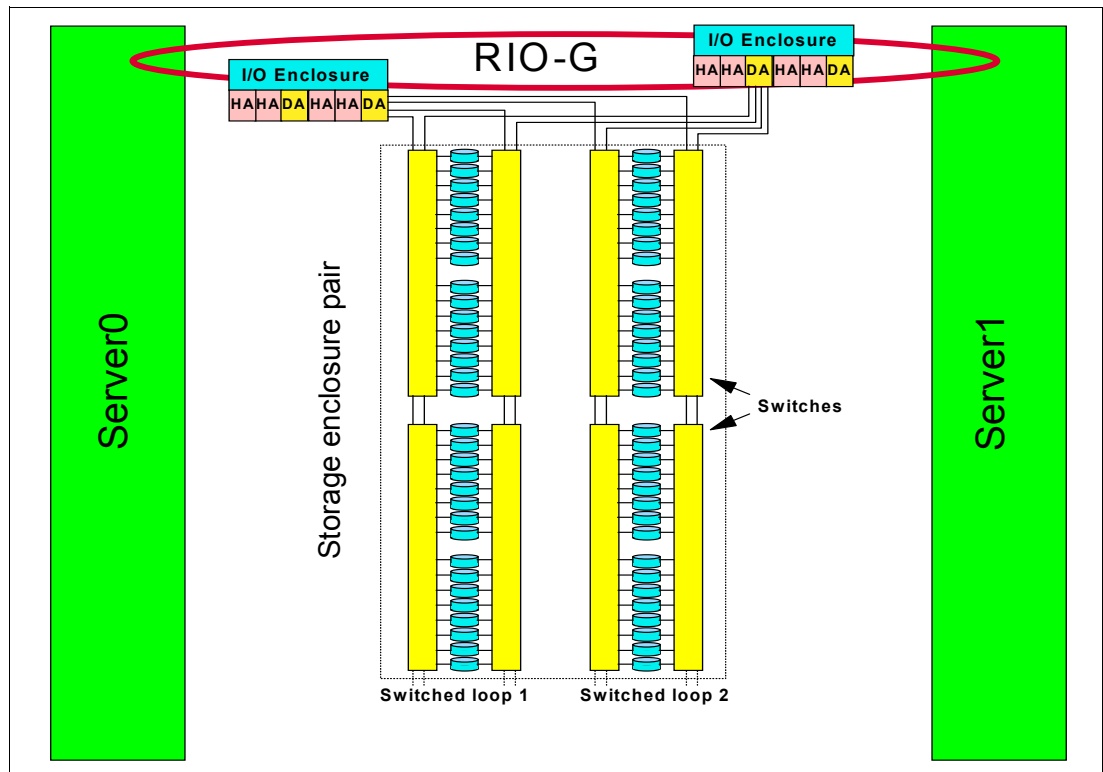


Figure 2-19 Physical layer as the base for virtualization

Compare this figure with the ESS design, where there was a real loop, and having an 8-pack close to a device adapter was an advantage. This design is no longer relevant for the DS8000. Because of the switching design, each drive is in close reach of the device adapter, apart from a few more hops through the Fibre Channel switches for some drives. So, it is not really a loop, but a switched FC-AL loop with the FC-AL addressing schema: Arbitrated Loop Physical Addressing (AL-PA).

### 2.9.3 Array sites

An *array site* is a group of eight DDMs. Although which DDMs are forming an array site is predetermined automatically by the DS8000, there is no predetermined server affinity for array sites. The DDMs selected for an array site are chosen from two disk enclosures on separate loops; see Figure 2-20 on page 49. The DDMs in the array site are of the same DDM type, which means the same capacity and the same speed (RPM).

As Figure 2-20 shows, array sites span loops. Four DDMs are taken from loop 1 and another four DDMs from loop 2. Array sites are the building blocks that are used to define arrays.

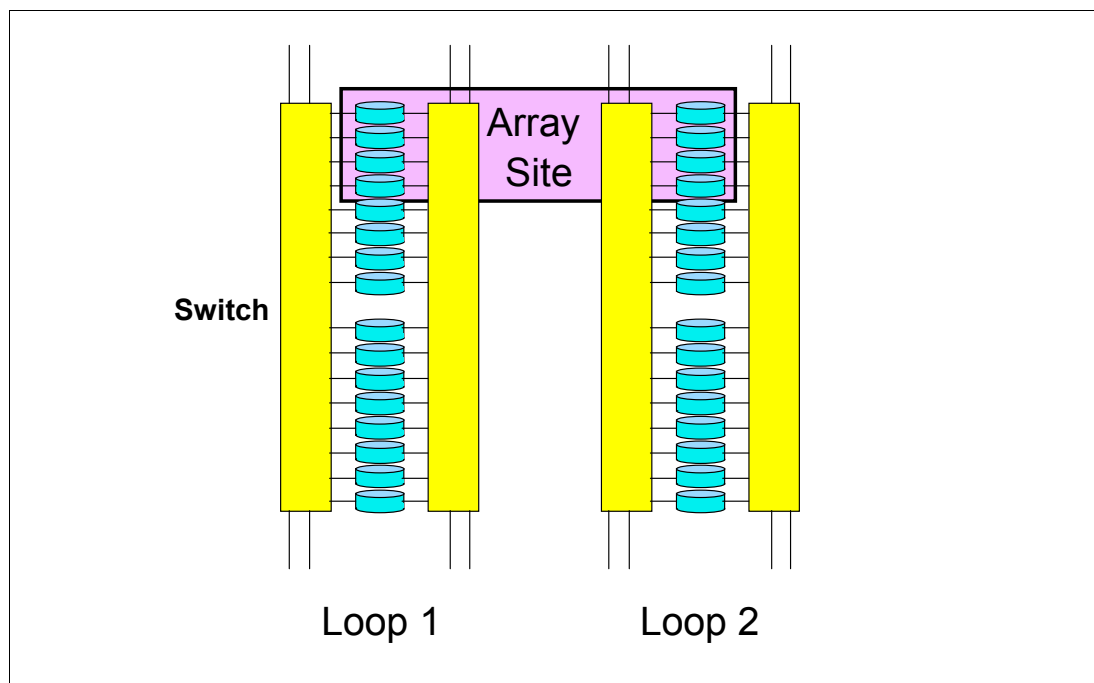


Figure 2-20 Array sites

## 2.9.4 Arrays

An *array* is created from one *array site*. Forming an array means defining it as a specific RAID type. The supported RAID types are RAID 5, RAID 6, and RAID 10 (see “RAID 5 implementation in the DS8000” on page 39, “RAID 6 implementation in the DS8000” on page 39, and “RAID 10 implementation in the DS8000” on page 40). For each array site, you can select a RAID type, with the one exception that solid-state drives can be configured *only* as RAID 5. The process of selecting the RAID type for an array is also called *defining* an array.

**Note:** In the DS8000 implementation, one array is defined using one array site.

According to the DS8000 sparing algorithm, from zero to two spares are taken automatically from the array site.

Figure 2-21 on page 50 shows the creation of a RAID 5 array with one spare, also called a 6+P+S array (capacity of 6 DDMs for data, capacity of one DDM for parity, and a spare drive). According to the RAID 5 rules, parity is distributed across all seven drives in this example.

On the right side in Figure 2-21, the terms D1, D2, D3, and so on stand for the set of data contained on one disk within a stripe on the array. If, for example, 1 GB of data is written, it is distributed across all the disks of the array.

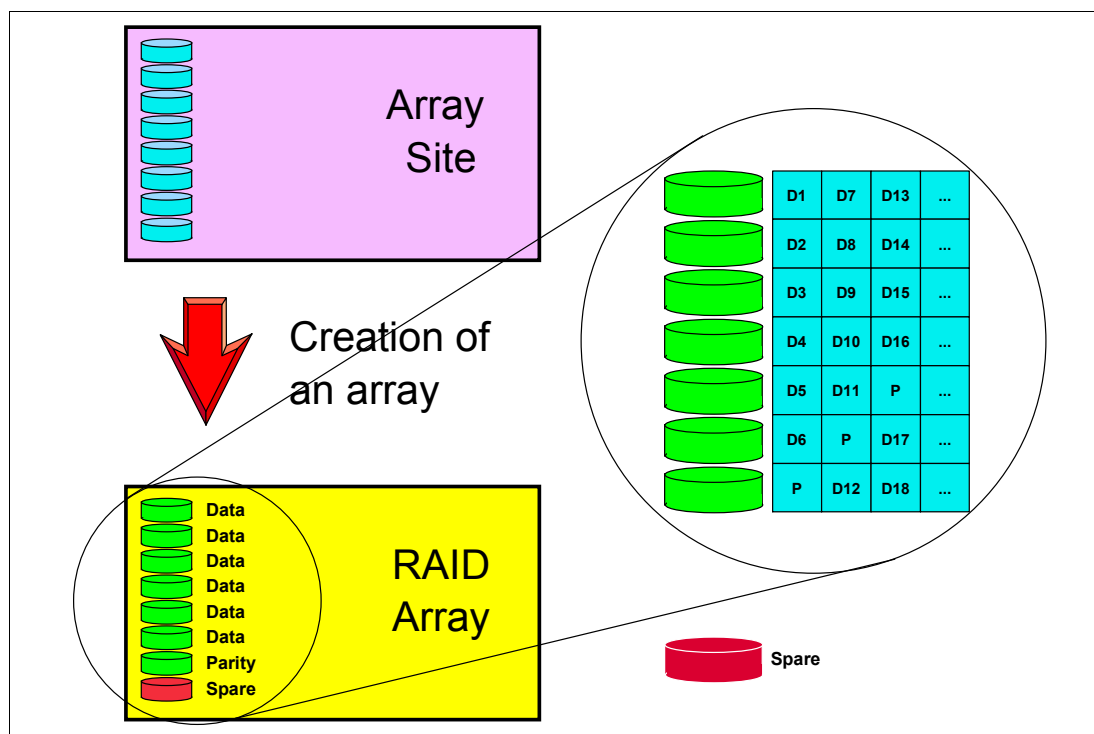


Figure 2-21 Creation of a array

Therefore, an array is formed using one array site. Although the array can be accessed by each adapter of the device-adapter pair, it is managed by one device adapter. You define which adapter and which server is managing this array later on in the configuration path.

## 2.9.5 Ranks

In the DS8000 virtualization hierarchy, there is another logical construct called a *rank*. When defining a new rank, its name is chosen by the DS Storage Manager, for example, R1, R2, or R3, and so on. You have to add an array to a rank.

**Note:** In the DS8000 implementation, a rank is built using only one array.

The available space on each rank is divided into *extents*. The extents are the building blocks of the logical volumes. An extent is striped across all disks of an array as shown in Figure 2-22 on page 51 and indicated by the small squares in Figure 2-23 on page 53.

The process of forming a rank does two tasks:

- ▶ The array is formatted for either fixed block (FB) data for open systems or count key data (CKD) for System z data. This task determines the size of the set of data contained on one disk within a stripe on the array.
- ▶ The capacity of the array is subdivided into equal-sized partitions, called *extents*. The extent size depends on the *extent type*, FB or CKD.



An FB rank has an extent size of 1 GB (more precisely, GiB, gibibyte, or binary gigabyte, being equal to  $2^{30}$  bytes).

System z users or administrators typically do not deal with gigabytes or gibibytes, and instead they think of storage in terms of the original 3390 volume sizes. A 3390 Model 3 is three times the size of a Model 1, and a Model 1 has 1113 cylinders, which is about 0.94 GB. The extent size of a CKD rank therefore was chosen to be one 3390 Model 1 or 1113 cylinders.

One extent is the minimum physical allocation unit when a CKD volume is created, as we discuss later. It is still possible to define a CKD volume with a capacity that is an integral multiple of one cylinder. However, if the defined capacity is not an integral multiple of the capacity of one extent, the unused capacity in the last extent is wasted. For example, you can define a one-cylinder CKD volume, but 1113 cylinders (1 extent) will be allocated and 1112 cylinders is wasted.

Figure 2-22 shows an example of an array that is formatted for FB data with 1 GB extents (the squares in the rank just indicate that the extent is composed of several blocks from separate DDMs).

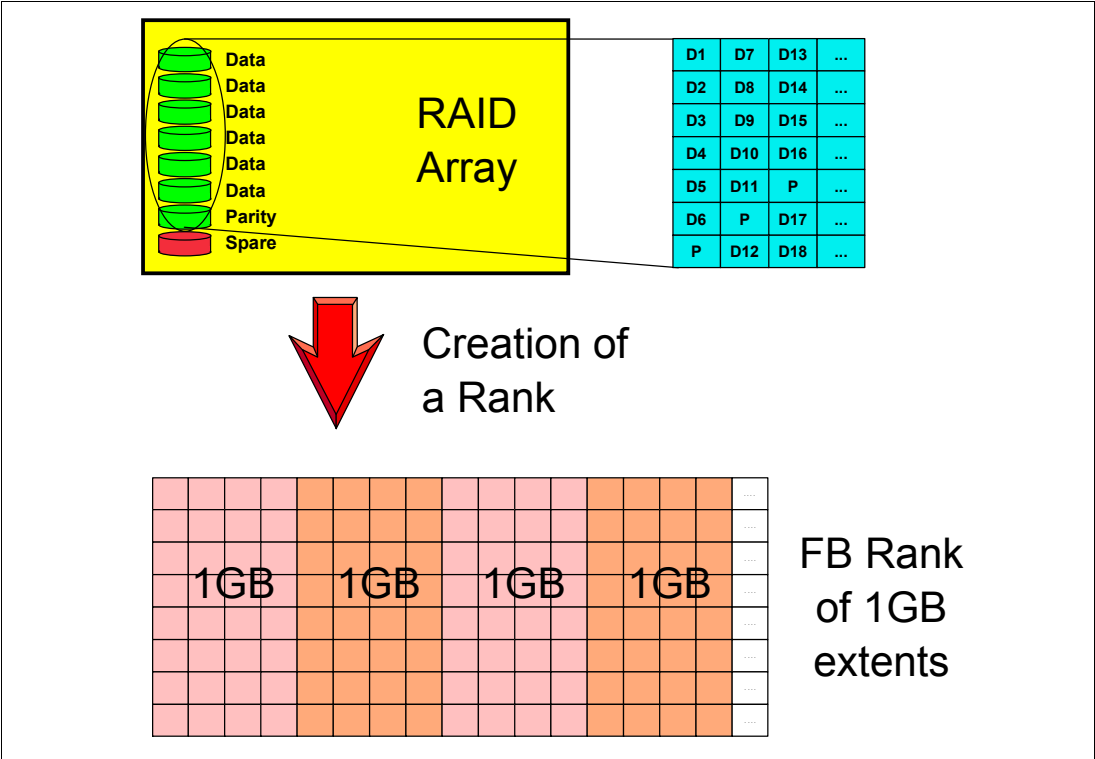


Figure 2-22 Forming an FB rank with 1 GB extents

### 2.9.6 Extent pools

An *extent pool* is a logical construct to aggregate the extents from a set of ranks, forming a domain for extent allocation to a logical volume. Typically, the set of ranks in the extent pool must have the same RAID type and the same disk RPM characteristics so that the extents in the extent pool have homogeneous characteristics.

**Important:** Do not mix ranks that have a different RAID type or disk RPM in an extent pool.

There is no predefined affinity of ranks or arrays to a storage server. The affinity of the rank (and its associated array) to a given server is determined at the point it is assigned to an extent pool.

One or more ranks *with the same extent type (FB or CKD)* can be assigned to an extent pool. One rank can be assigned to only one extent pool. There can be as many extent pools as there are ranks.

Several considerations exist regarding how many ranks can be added in an extent pool.

*Storage Pool Striping* was made available with Licensed Machine Code 5.3.xx.xx and allows you to create logical volumes striped across multiple ranks. This technique can typically enhance performance. To benefit from Storage Pool Striping (see “Rotate volumes allocation method” on page 55), more than one rank in an extent pool is required.

Although Storage Pool Striping can significantly enhance performance, when you lose one rank (in the unlikely event that a whole RAID array failed because of a scenario with multiple failures at the same time), the data of this rank is lost, and so is all the data in this extent pool, because data is striped across all ranks. Therefore, keep the number of ranks in an extent pool in the range of four to eight.

**Tip:** Use four to eight ranks with the same characteristics in an extent pool.

The DS Storage Manager GUI guides you to use the same RAID types in an extent pool. As such, when an extent pool is defined, it must be assigned with the following attributes:

- ▶ Server affinity
- ▶ Extent type
- ▶ RAID type
- ▶ Drive class

The minimum number of extent pools is two, with one assigned to server 0 and the other to server 1 so that both servers are active. In an environment where FB and CKD are to go onto the DS8000 storage server, four extent pools can provide one FB pool for each server, and one CKD pool for each server, to balance the capacity between the two servers.

Figure 2-23 on page 53 is an example of a mixed environment with CKD and FB extent pools. Additional extent pools might also be desirable to segregate ranks with various DDM types. Extent pools are expanded by adding more ranks to the pool. Ranks are organized in two *rank groups*; rank group 0 is controlled by server 0 and rank group 1 is controlled by server 1.

**Important:** Capacity should be balanced between the two servers for best performance.

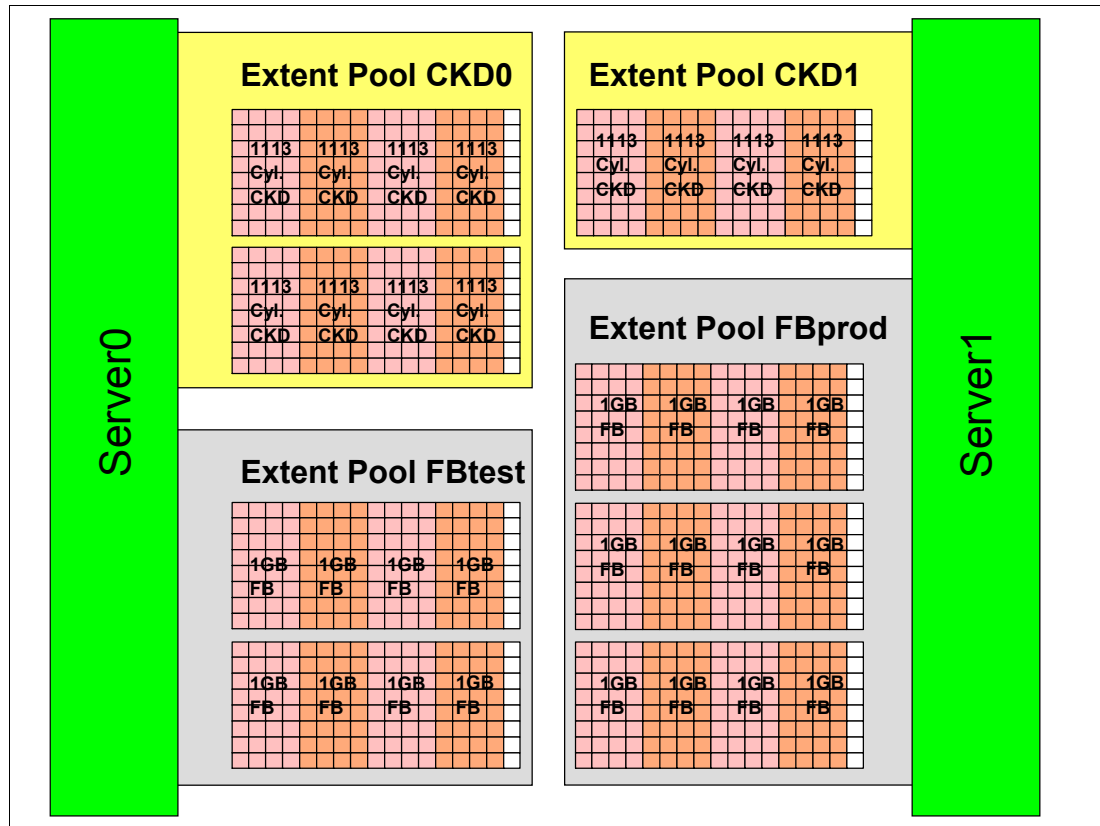


Figure 2-23 Extent pools

## 2.9.7 Logical volumes

A *logical volume* consists of a set of extents from one extent pool.

On a DS8000, up to 65280 volumes can be created (either 64K CKD, or 64K FB volumes, or a mixture of both types with a maximum of 64K volumes in total).

**Abbreviation for 64K:** For 65280, we use the abbreviation 64K in this discussion, even though it is actually 65536 minus 256, which is not quite 64K in binary.

### CKD volumes

A System z CKD volume consists of one or more extents from one CKD extent pool. CKD extents are of the size of 3390 Model 1, which has 1113 cylinders. However, when you define a System z CKD volume, you do not specify the number of 3390 Model 1 extents but the number of cylinders you want for the volume.

Prior to Licensed Machine Code 5.4.0xx.xx, the maximum size for a CKD volume was 65,520 cylinders. Now you can define CKD volumes with up to 262,668 cylinders, which is approximately 223 GB. This new volume capacity is called extended address volume (EAV) and the device type is 3390 Model A.

**Important:**

- ▶ EAV volumes can be used only by z/OS 1.10 or later versions.
- ▶ Thin Provisioning of volumes, as included in Licensed Machine Code 5.4.3xx.xx, is not supported for CKD volumes at this time.

If the number of cylinders specified is not an exact multiple of 1113 cylinders, some space in the last allocated extent is wasted. For example, if you define 1114 or 3340 cylinders, 1112 cylinders are wasted. For maximum storage efficiency, consider allocating volumes that are exact multiples of 1113 cylinders. In fact, consider multiples of 3339 cylinders for future compatibility.

If you want to use the maximum number of cylinders for a volume on a DS8000 running Licensed Machine Code 5.4.xx.xx (that is 262,668 cylinders), you are not wasting cylinders, because it is an exact multiple of 1113 (262,668 divided by 1113 is exactly 236). For even better future compatibility, you should use a size of 260,442 cylinders, which is an exact multiple (78) of 3339, a model 3 size. On DS8000s running previous Licensed Machine Codes, the maximum number of cylinders is 65,520 and it is *not* a multiple of 1113. You may go with 65,520 cylinders and waste 147 cylinders for each volume (the difference to the next multiple of 1113) or you might be better with a volume size of 64,554 cylinders, which is a multiple of 1113 (factor of 58), or even better, with 63,441 cylinders, which is a multiple of 3339, a model 3 size. See Figure 2-24.

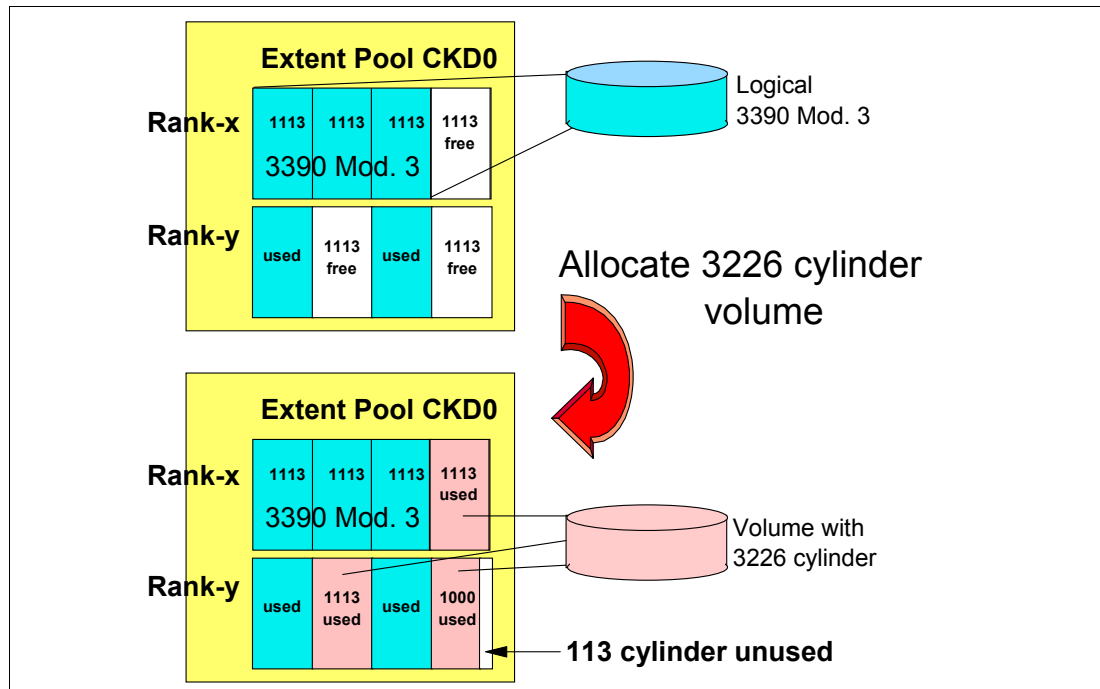


Figure 2-24 Allocation of a CKD logical volume

A CKD volume cannot span multiple extent pools, but a volume can have extents from separate ranks in the same extent pool or you can stripe a volume across the ranks (see “Rotate volumes allocation method” on page 55). Figure 2-24 shows how a logical volume is allocated with a CKD volume as an example.

## 2.9.8 Allocation, deletion, and modification of CKD volumes

All extents of the ranks assigned to an extent pool are independently available for allocation to logical volumes. The extents for a volume are logically ordered, but they do not have to come from one rank and the extents do not have to be contiguous on a rank.

This construction method of using fixed extents to form a logical volume in the DS8000 allows flexibility in the management of the logical volumes. We can delete CKD volumes, resize volumes, and reuse the extents of those volumes to create other volumes, maybe of different sizes. One logical volume can be removed without affecting the other logical volumes defined on the same extent pool.

Because the extents are *cleaned* after you have deleted a CKD volume, it can take some time until these extents are available for reallocation. The reformatting of the extents is a background process.

The two extent allocation algorithms of the DS8000 are *rotate volumes* and *Storage Pool Striping (rotate extents)*.

### Rotate volumes allocation method

Extents can be allocated sequentially. In this case all extents are taken from the same rank until we have enough extents for the requested volume size or the rank is full, in which case the allocation continues with the next rank in the extent pool.

If more than one volume is created in one operation, the allocation for each volume starts in another rank. When allocating several volumes, we *rotate* through the ranks.

You might want to consider this allocation method when you prefer to manage performance manually. Because the workload of one volume is going to one rank, the identification of performance bottlenecks is easier. However, by putting all the volumes data onto only one rank, you might introduce a bottleneck, depending on your actual workload.

### Storage Pool Striping: rotate extents

The second and actually preferred storage allocation method is *Storage Pool Striping*. Storage Pool Striping is an option (introduced with Licensed Machine Code 5.3.x.xxx) when a volume is created. The extents of a volume can be striped across several ranks.

To use this storage allocation method, it is obvious that you need an extent pool with more than one rank.

The DS8000 keeps a sequence of ranks. The first rank in the list is randomly picked at each power-on of the storage subsystem. The DS8000 keeps track of the rank in which the last allocation started. The allocation of a first extent for the next volume starts from the next rank in that sequence. The next extent for that volume is taken from the next rank in sequence and so on. Therefore, the system rotates the extents across the ranks.

Figure 2-25 shows an example of how volumes get allocated within the extent pool.

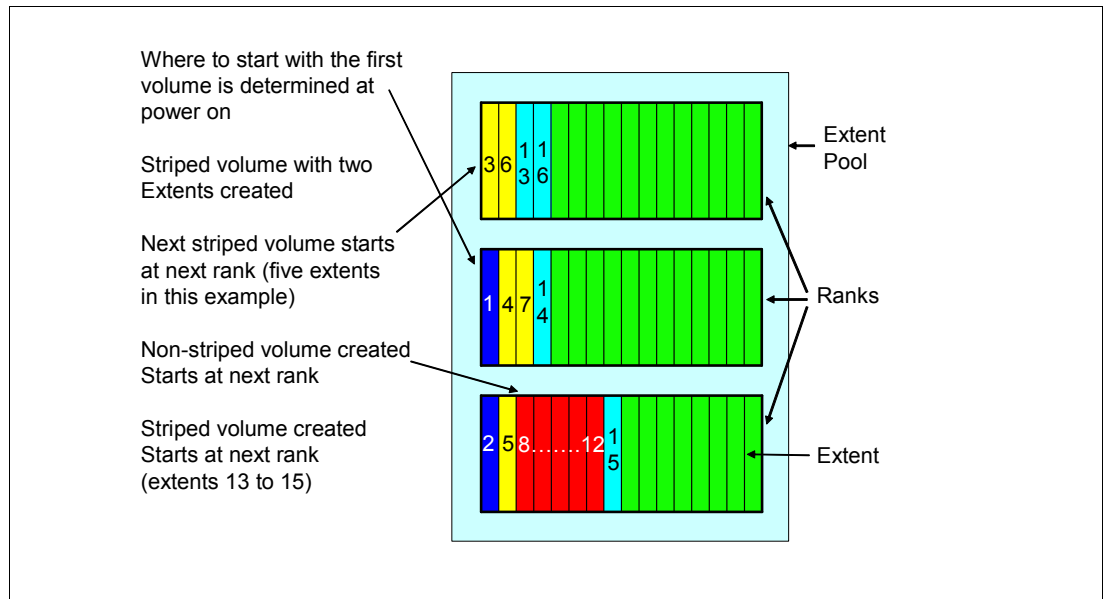


Figure 2-25 Extent allocation methods

When you create striped volumes and non-striped volumes in an extent pool, a rank might be filled before the others. A full rank is skipped when you create new striped volumes.

No reorganization function exists for the extents in an extent pool. If you add one or more ranks to an existing extent pool, the existing extents are not redistributed.

**Tip:** If you have to add capacity to an extent pool because it is nearly full, a better solution is to add several ranks at once instead of only one rank. This solution allows new volumes to be striped across the added ranks.

Figure 2-26 on page 57 shows both standard volumes and extent space-efficient (ESE) volumes in an extent pool, both using the rotate volumes allocation method (solid colors show standard volumes 8 - 12, and striped colors show ESE volumes 17 - 19, 26 - 27) and Storage Pool Striping extent allocation methods (all other standard and ESE volumes).

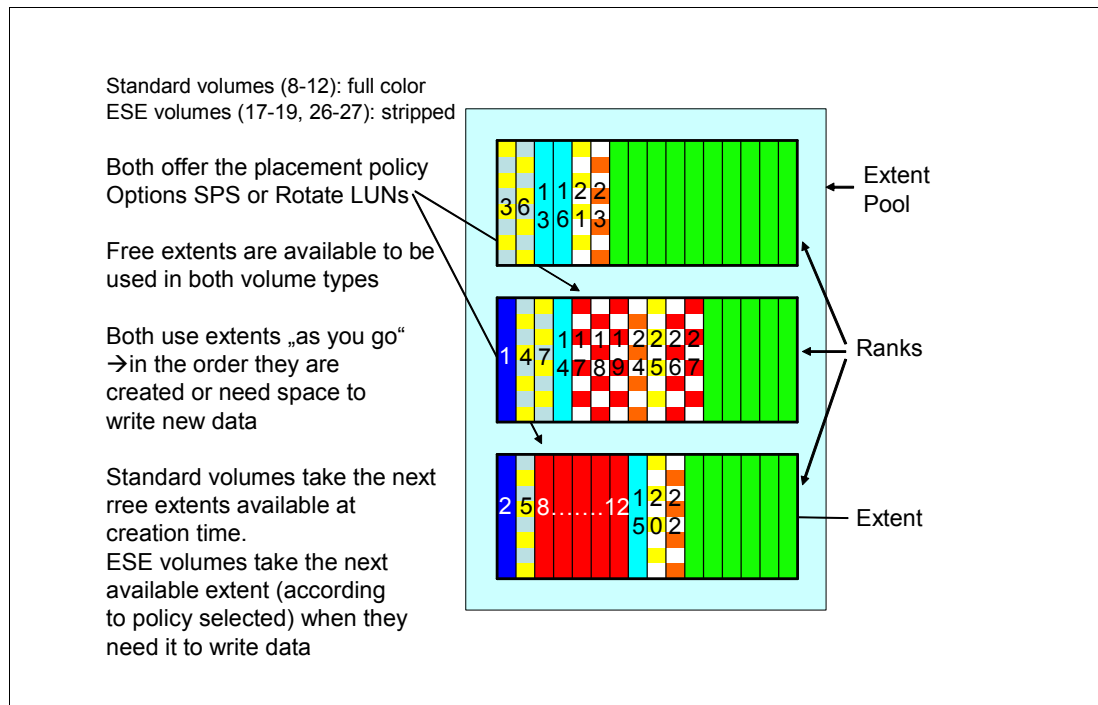


Figure 2-26 Free extents can be used for standard and extent space-efficient volumes

Note the following usage information:

► Use of extent space-efficient (ESE) volumes

Like standard volumes (which are fully provisioned), ESE volumes can be mapped to hosts. They are supported in combination with Copy Services functions starting with z/OS V1.12.

► Use of track space-efficient (TSE) volumes

TSE volumes are supported as FlashCopy target volumes only.

By using striped volumes, you distribute the I/O load to a CKD volume to more than only one set of eight disk drives. The ability to distribute a workload to many physical drives can greatly enhance performance for a logical volume. In particular, operating systems that do not have a volume manager that can do striping can benefit most from this allocation method.

However, if you have extent pools with many ranks and all volumes are striped across the ranks and you lose only one rank (for example, because there are two disk drives in the same rank that fail at the same time and it is not a RAID 6 rank), you will lose a lot of your data. Therefore, having extent pools with only approximately four to eight ranks might be better.

If you decide to use Storage Pool Striping, a better might be to use this allocation method for all volumes in the extent pool to keep the ranks equally filled and utilized.

**Tip:** If you configure a new DS8000, do not mix volumes using the Storage Pool Striping method and volumes using the rotate volumes method in the same extent pool.

## 2.9.9 Data striping by access method

Usually, in a multi-extent, multi-volume VSAM data set that is processed in sequential access mode, processing does not present any type of parallelism for I/O operations among the

volumes. This statement means that when an I/O operation is executed for an extent in a volume, no other I/O activity from the same task or same data set is scheduled to the other volumes. In a situation where I/O is the major bottleneck, and there are available resources in the channel subsystem and controllers, these resources can be wasted.

Data striping addresses this performance problem by imposing two modifications to the traditional data organization:

- ▶ The records are not placed in *key ranges* along the volumes; instead they are organized in stripes.
- ▶ Parallel I/O operations are scheduled to sequential stripes in separate volumes.

By striping data, the tracks in the case of Sequential Access Method (SAM), and the control intervals (CIs) for VSAM, are spread across multiple devices. This format allows a single application request for records in multiple tracks and CIs to be satisfied by concurrent I/O requests to multiple volumes.

The result is improved performance by achieving data transfer into the application at a rate greater than any single I/O path.

### **VSAM data striping**

Data striping support for VSAM was initially provided with DFSMS 2.10. Support is provided for all VSAM organization, including the following types:

- ▶ Key-sequenced data set (KSDS)
- ▶ Entry-sequenced data set (ESDS)
- ▶ Relative record data set (RRDS)
- ▶ variable relative record data set (VRRDS)
- ▶ Linear data set (LDS)

DB2 uses linear VSAM data sets (LDS) for its table spaces. All the control (including buffer pool) is done by DB2. For example, DB2 implements data striping in LDS.

VSAM data striping allows sequential I/O to be performed for a data set at a rate greater than that allowed by the physical path between the disk and the processor. The physical characteristics of channels, control units, and disks limit the data transfer rate. VSAM data striping avoids such limitations by spreading the data set among multiple stripes on multiple control units. This approach differs from most of the RAID technologies that stripe data in the same disk array. The data striping function is designed to improve the performance of applications requiring sequential access to data records. Data striping does not affect direct access to data.

An equal amount of space is allocated for each stripe. If the guaranteed space attribute is used in the storage class, the specified quantity is allocated to each volume in the stripe count. For a data set with the non-guaranteed space attribute in the storage class, the initial allocation quantity is divided across all volumes in the stripe count.

Striped VSAM data sets are in extended format (EF) and are internally organized so that CIs are distributed across a group of disk volumes or stripes. A CI is contained within a stripe.

DB2 V8 enables all pages sizes to be striped when DSNZPARM value for DSVCI is set to YES. For issues with pages greater than 4 KB prior to DB2 V8, see APAR PQ53571.

VSAM data striping should be used only for those non-partitioned objects where the majority of activity is heavy sequential processing.



Because of their highly sequential nature, use VSAM data striping in the following locations:

- ▶ DB2 active log data sets
- ▶ DB2 work files
- ▶ Segmented table spaces and non-partitioned indexes (NPIs) that are mostly and heavily sequential
- ▶ LOBs

### Sequential data striping

In addition to being beneficial to stripe an LDS that is heavily sequential, another benefit is to stripe, when possible, the input and output files for utilities that are associated with a striped table space. The following examples are what can be beneficial to stripe when the associated table space is striped:

- ▶ Image copy data sets
- ▶ Input to LOADs
- ▶ Utility work files

Only disk data sets can be striped.

## 2.9.10 Dynamic volume expansion

The size of a CKD volume can be expanded without destroying the data. On the DS8000 side, we simply add extents to the volume. Of course, the operating system has to support this resizing.

VTOC, VTOC index, and VSAM volume data set (VVDS) must be large enough to handle the increase in extents.

**Note:** A possibility is to expand a 3390 Model 9 volume to a 3390 Model A volume. This possibility means that when executing Dynamic Volume Expansion on a 3390 Model 9, and the new capacity is bigger than 65,520 cylinders, the system will enlarge the volume and change its *data type* to 3390-A.

A logical volume has the attribute of being striped across the ranks or not. If the volume was created as striped across the ranks of the extent pool, the extents that are used to increase the size of the volume are also striped. If a volume was created without striping, the system tries to allocate the additional extents within the same rank that the volume was created from originally.

Because most operating systems have no means to move data from the end of the *physical* disk off to some unused space at the beginning of the disk, it makes no sense to *reduce* the size of a volume. The DS8000 configuration interfaces, DS CLI and DS GUI, do *not* allow you to change a volume to a smaller size.

**Important:** Before you can expand a volume you have to delete any copy services relationship involving that volume.

## 2.9.11 Logical subsystems

A *logical subsystem* (LSS) is another logical construct, grouping logical volumes into groups of up to 256 logical volumes. You can define up to 255 LSSs for the DS8000. You can even have more LSSs than arrays.

Previously, on an ESS, there was a fixed association between logical subsystems (and their associated logical volumes) and device adapters (and their associated ranks). The association of an eight-pack to a device adapter determined what LSS numbers could be chosen for a volume. On an ESS, up to 16 LSSs could be defined depending on the physical configuration of device adapters and arrays.

On the DS8000, there is no fixed binding between any rank and any logical subsystem. The capacity of one or more ranks can be aggregated into an extent pool, and logical volumes that are configured in that extent pool are not bound to any specific rank. Different logical volumes on the same logical subsystem can be configured in different extent pools. As such, the available capacity of the storage facility can be flexibly allocated across the set of defined logical subsystems and logical volumes.

This predetermined association between array and LSS is gone on the DS8000. Also, the number of LSSs has changed. You can now define up to 255 LSSs for the DS8000. You can even have more LSSs than arrays.

For each CKD volume, you can choose an LSS. Remember, you can put up to 256 volumes into one LSS. There is, however, one restriction. We already have seen that volumes are formed from a bunch of extents from an extent pool. extent pools, however, belong to one server, server 0 or server 1, respectively. Therefore, LSSs also have an affinity to the servers. All even-numbered LSSs (X'00', X'02', X'04', up to X'FE') belong to server 0 and all odd-numbered LSSs (X'01', X'03', X'05', up to X'FD') belong to server 1. LSS X'FF' is reserved.

System z users are familiar with a *logical control unit* (LCU). System z operating systems configure LCUs to create device addresses. There is a one-to-one relationship between an LCU and a CKD LSS (LSS X'ab' maps to LCU X'ab'). Logical volumes have a logical volume number X'abcd' where X'ab' identifies the LSS and X'cd' is one of the 256 logical volumes on the LSS. This logical volume number is assigned to a logical volume when a logical volume is created and determines the LSS that it is associated with. The 256 possible logical volumes associated with an LSS are mapped to the 256 possible device addresses on an LCU (logical volume X'abcd' maps to device address X'cd' on LCU X'ab'). When creating CKD logical volumes and assigning their logical volume numbers, consider whether parallel access volumes (PAV) are required on the LCU and reserve some of the addresses on the LCU for alias addresses.

Certain management actions in Metro Mirror, Global Mirror, or Global Copy operate at the LSS level. For example, the freezing of pairs to preserve data consistency across all pairs, in case you have a problem with one of the pairs, is done at the LSS level. With the option to put all or most of the volumes of a certain application in only one LSS, the management of remote copy operations is easier. See Figure 2-27 on page 61.

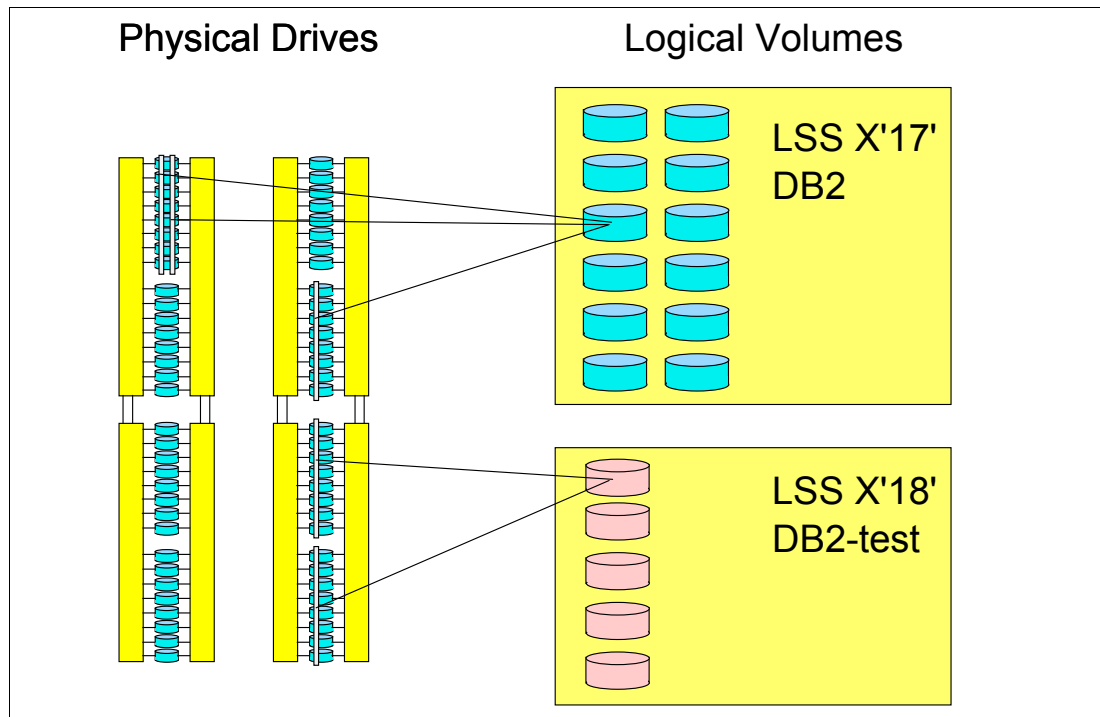


Figure 2-27 Grouping of volumes in LSSs

Previously, of course, you could have put all volumes for one application in one LSS on an ESS, too, but then all volumes of that application would also have been in one or a few arrays, and from a performance standpoint, this was not desirable. Now on the DS8000, you can group your volumes in one or a few LSSs but still have the volumes in many arrays or ranks.

Fixed block LSSs are created automatically when the first fixed block logical volume on the LSS is created, and deleted automatically when the last fixed block logical volume on the LSS is deleted. CKD LSSs require user parameters to be specified and must be created before the first CKD logical volume can be created on the LSS; they must be deleted manually after the last CKD logical volume on the LSS is deleted.

### Address groups

Address groups are created automatically when the first LSS that is associated with the address group is created, and deleted automatically when the last LSS in the address group is deleted.

LSSs are either CKD LSSs or FB LSSs. All devices in an LSS must be either CKD *or* FB. This restriction goes even further. LSSs are grouped into address groups of 16 LSSs. LSSs are numbered X'*ab*', where *a* is the address group and *b* denotes an LSS within the address group. So, for example, X'10' to X'1F' are LSSs in address group 1.

All LSSs within one address group have to be of the same type, CKD or FB. The first LSS defined in an address group fixes the type of that address group.

**Important:** System z users who still want to use ESCON to attach hosts to the DS8000 must be aware that ESCON supports only the 16 LSSs of address group 0 (LSS X'00' to X'0F'). Therefore, this address group must be reserved for ESCON-attached CKD devices in this case, and not used as FB LSSs.

Figure 2-28 shows the concept of LSSs and address groups.

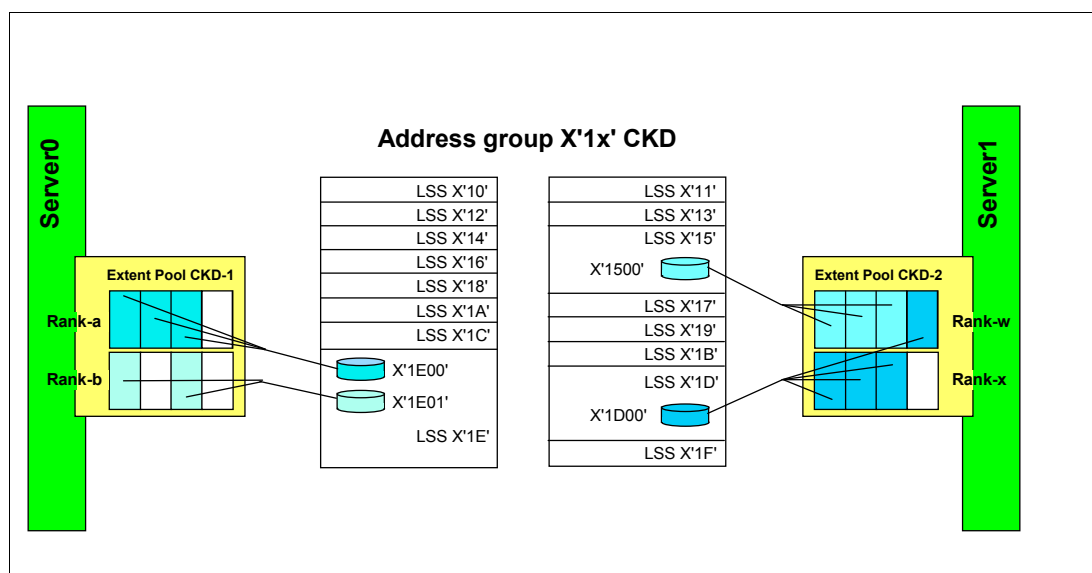


Figure 2-28 Logical storage subsystem

## Volume groups

A *volume group* is a named construct that defines a set of logical volumes. When used in conjunction with CKD hosts, there is a default volume group that contains all CKD volumes and any CKD host that logs in to a FICON I/O port has access to the volumes in this volume group. CKD logical volumes are automatically added to this volume group when they are created, and are automatically removed from this volume group when they are deleted.

## 2.9.12 Summary of the virtualization hierarchy

The virtualization hierarchy is summarized as follows:

1. We start with a bunch of disks that are grouped in array sites.
2. An array site is transformed into an array, eventually with spare disks.
3. The array is further transformed into a rank with extents, formatted for FB data or CKD.
4. The extents are added to an extent pool that determines which storage server serves the ranks, and aggregates the extents of all ranks in the extent pool for subsequent allocation to one or more logical volumes. Within the extent pool, we can reserve some space for TSE volumes by means of creating a repository. Both ESE and TSE volumes require virtual capacity to be available in the extent pool.
5. We create logical volumes within the extent pools (optionally striping the volumes), assigning them a logical volume number that determines which logical subsystem they are to be associated with and which server can manage them. This is the same for both Standard volumes (fully allocated) and extent space-efficient volumes. TSE volumes for use with FlashCopy SE can be created only within the repository of the extent pool.

This virtualization concept provides much more flexibility than in previous products. Logical volumes can dynamically be created, deleted, and resized. They can be grouped logically to simplify storage management. Large CKD volumes reduce the total number of volumes, which also contributes to a reduction of the management effort.

Figure 2-29 summarizes the virtualization hierarchy.

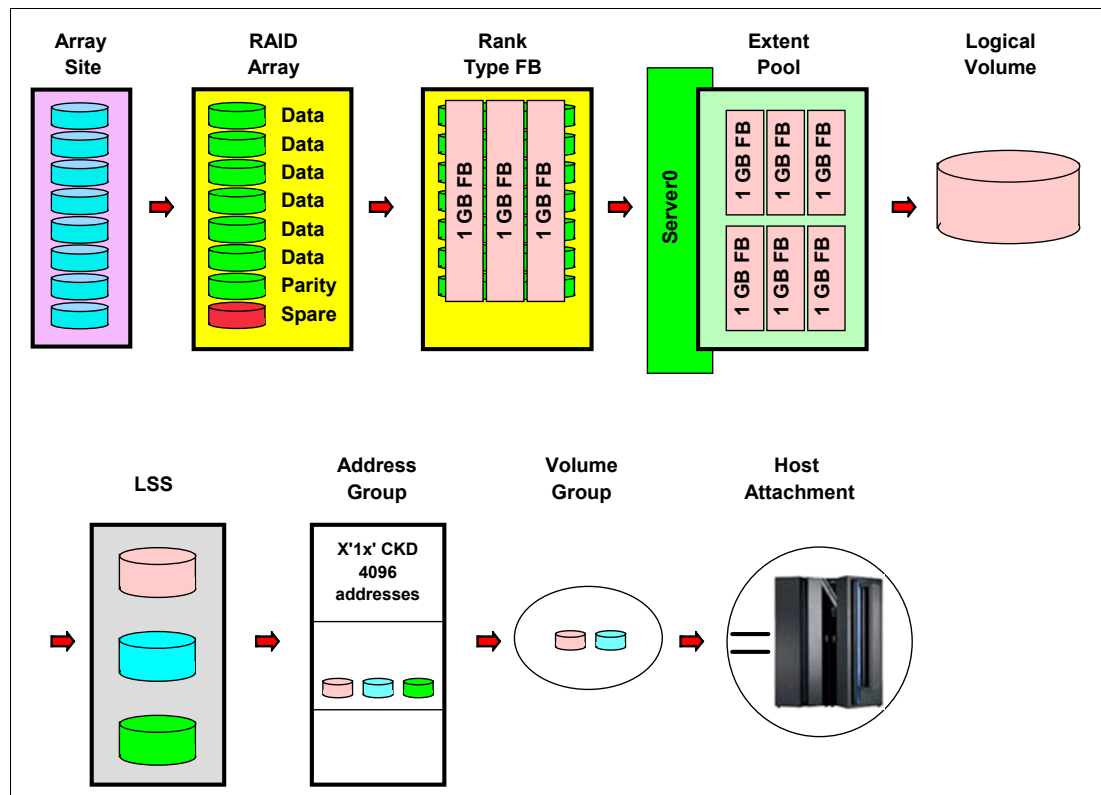


Figure 2-29 Virtualization hierarchy

### 2.9.13 Benefits of virtualization

The DS8000 physical and logical architecture defines new standards for enterprise storage virtualization. The main benefits of the virtualization layers are as follows:

- ▶ Flexible LSS definition allows maximization and optimization of the number of devices for each LSS.
- ▶ No strict relationship exists between RAID ranks and LSSs.
- ▶ No connection of LSS performance to underlying storage exists.
- ▶ Number of LSSs can be defined, based upon device number requirements:
  - With larger devices, significantly fewer LSSs might be used.
  - Volumes for a particular application can be kept in a single LSS.
  - Smaller LSSs can be defined if required (for applications requiring less storage).
  - Test systems can have their own LSSs with fewer volumes than production systems.
- ▶ Increased number of logical volumes exists:
  - Up to 65280 (CKD)
  - Up to 65280 (FB)
  - 65280 total for CKD + FB
- ▶ Any mixture of CKD or FB addresses in 4096 address groups is allowed.
- ▶ Logical volume size is increased:
  - CKD: 223 GB (262,668 cylinders), architecture for 219 TB
  - FB: 2 TB, architecture for 1 PB

- ▶ Logical volume configuration is flexible:
  - Multiple RAID types (RAID 5, RAID 6, and RAID 10)
  - Storage types (CKD and FB) aggregated into extent pools
  - Volumes allocated from extents of extent pool
  - Storage pool striping
  - Dynamically add and remove volumes
  - Dynamic volume expansion
  - Extent space-efficient volumes for thin provisioning
  - Track space-efficient volumes for FlashCopy SE
  - Extended address volumes (CKD)
- ▶ Virtualization reduces storage management requirements.

## 2.10 Caching algorithms in DS8000

Most, if not all, high-end disk systems have internal cache integrated into the system design, and some amount of system cache is required for operation. Although cache sizes have dramatically increased over time, the ratio of cache size to system disk capacity has remained nearly the same. Caching algorithms have been improved to adapt with resources that are available and access profiles.

In this section, we discuss the following algorithms:

- ▶ Sequential adaptive replacement cache (SARC)
- ▶ Adaptive Multi-stream Prefetching (AMP) (AMP)
- ▶ Intelligent Write Caching

### 2.10.1 Sequential adaptive replacement cache (SARC)

The DS8000 uses the SARC algorithm, which was developed by IBM Storage Development in partnership with IBM Research. It is a self-tuning, self-optimizing solution for a wide range of workloads with a varying mix of sequential and random I/O streams. SARC is inspired by the adaptive replacement cache (ARC) algorithm and inherits many features from it<sup>5</sup>.

SARC basically attempts to determine the following information:

- ▶ When data is copied into the cache.
- ▶ Which data is copied into the cache.
- ▶ Which data is evicted when the cache becomes full.
- ▶ How the algorithm dynamically adapts to various workloads.

The DS8000 cache is organized in 4-KB pages called *cache pages* or *slots*. This unit of allocation (which is smaller than the values used in other storage systems) ensures that small I/Os do not waste cache memory.

The decision to copy some amount of data into the DS8000 cache can be triggered from two policies: demand paging and prefetching. *Demand paging* means that eight disk blocks (a 4K cache page) are brought in only on a cache miss. Demand paging is always active for all volumes and ensures that I/O patterns with some locality find at least some recently used data in the cache. *Prefetching* means that data is copied into the cache speculatively even before it is requested. To prefetch, a prediction of likely future data accesses is needed.

<sup>5</sup> For a detailed description of AMP, see *Outperforming LRU with an adaptive replacement cache algorithm*, by N. Megiddo and D.S. Modha, D. S. in IEEE Computer, volume 37, number 4, pages 58–65, 2004. For a detailed description of SARC, see *SARC: Sequential Prefetching in Adaptive Replacement Cache*, by Binny Gill and Dharmendra S. Modha in the Proceedings of the USENIX 2005 Annual Technical Conference, pages 293-308.

Because effective, sophisticated prediction schemes need extensive history of page accesses (which is not feasible in real systems), SARC uses prefetching for sequential workloads. Sequential access patterns naturally arise in video-on-demand, database scans, copy, backup, and recovery. The goal of sequential prefetching is to detect sequential access and effectively preload the cache with data so as to minimize cache misses. Today, prefetching is ubiquitously applied in web servers and clients, databases, file servers, on-disk caches, and multimedia servers.

For prefetching, the cache management uses tracks. A track is a set of 128 disk blocks (16 cache pages). To detect a sequential access pattern, counters are maintained with every track to record whether a track has been accessed together with its predecessor. Sequential prefetching becomes active only when these counters suggest a sequential access pattern. In this manner, the DS8000 monitors application read-I/O patterns and dynamically determines what to stage in the cache:

- ▶ Only the page requested
- ▶ That requested page plus remaining data on the disk track
- ▶ An entire disk track (or a set of disk tracks), which has not yet been requested

The decision of when and what to prefetch is made in accordance with the Adaptive Multi-stream Prefetching (AMP) algorithm, which dynamically adapts the amount and timing of prefetches optimally on a per-application basis (rather than a system-wide basis). AMP is described further in 2.10.2, “Adaptive Multi-stream Prefetching (AMP)” on page 66.

To decide which pages are evicted when the cache is full, sequential and random (non-sequential) data is separated into lists. Figure 2-30 illustrates the SARC algorithm for random and sequential data.

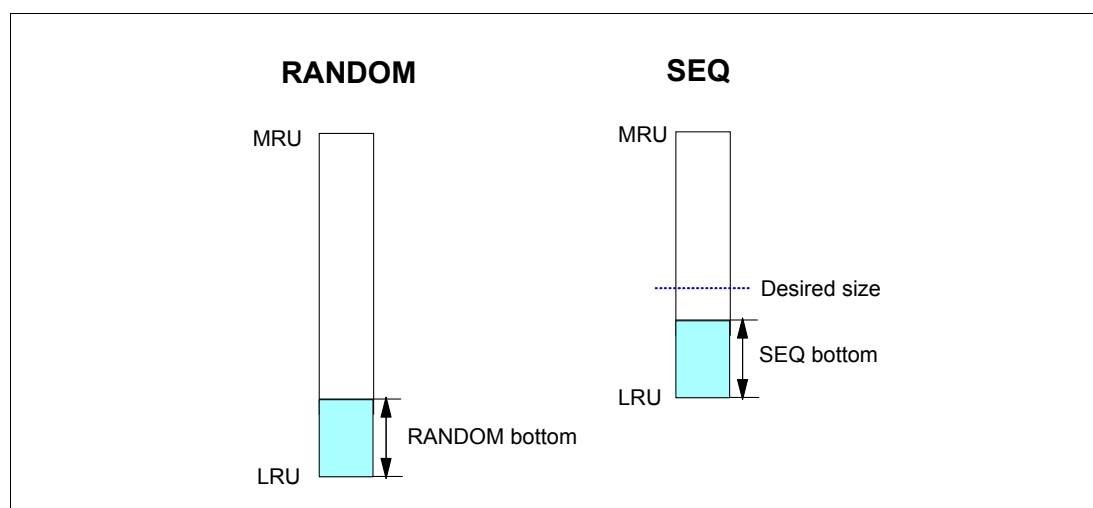


Figure 2-30 Sequential Adaptive Replacement Cache

A page that has been brought into the cache by simple demand paging is added to the most recently used (MRU) head of the RANDOM list. Without further I/O access, it goes down to the least recently used (LRU) bottom. A page that has been brought into the cache by a sequential access or by sequential prefetching is added to the MRU head of the SEQ list and then goes in that list. Additional rules control the migration of pages between the lists so that the same pages are not kept in memory twice.

To follow workload changes, the algorithm trades cache space between the RANDOM and SEQ lists dynamically and adoptively. This technique makes SARC scan-resistant, so that one-time sequential requests do not pollute the whole cache. SARC maintains a desired-size

parameter for the sequential list. The desired size is continually adapted in response to the workload. Specifically, if the bottom portion of the SEQ list is found to be more valuable than the bottom portion of the RANDOM list, the desired size is increased; otherwise, the desired size is decreased. The constant adaptation strives to make optimal use of limited cache space and delivers greater throughput and faster response times for a given cache size.

Additionally, the algorithm dynamically modifies not only the sizes of the two lists, but also the rate at which the sizes are adapted. In a steady state, pages are evicted from the cache at the rate of cache misses. A larger (respectively, a smaller) rate of misses effects a faster (respectively, a slower) rate of adaptation.

Other implementation details take into account the relationship of read and write (NVS) cache, efficient destaging, and the cooperation with Copy Services. In this manner, the DS8000 cache management goes far beyond the usual variants of the least recently used/least frequently used (LRU/LFU) approaches.

## 2.10.2 Adaptive Multi-stream Prefetching (AMP)

As described previously, SARC dynamically divides the cache between the RANDOM and SEQ lists, where the SEQ list maintains pages brought into the cache by sequential access or sequential prefetching.

Starting with Licensed Internal Code V5.2.400.327, the SEQ list is managed by a technology from IBM Research called *Adaptive Multi-stream Prefetching (AMP)*. AMP introduces an autonomic, workload-responsive, self-optimizing prefetching technology that adapts both the amount of prefetch and the timing of prefetch on a per-application basis in order to maximize the performance of the system.

The AMP algorithm solves two problems that plague most other prefetching algorithms:

- ▶ *Prefetch wastage* occurs when prefetched data is evicted from the cache before it can be used.
- ▶ *Cache pollution* occurs when less useful data is prefetched instead of more useful data.

By wisely choosing the prefetching parameters, AMP provides provably optimal sequential-read performance, maximizing the aggregate sequential-read throughput of the system. The amount prefetched for each stream is dynamically adapted according to the application's needs and the space available in the SEQ list. The timing of the prefetches is also continuously adapted for each stream to avoid misses and at the same time avoid any cache pollution.

SARC and AMP play complementary roles. While SARC is carefully dividing the cache between the RANDOM and the SEQ lists so as to maximize the overall hit ratio, AMP is managing the contents of the SEQ list to maximize the throughput obtained for the sequential workloads. SARC affects cases that involve both random and sequential workloads; AMP helps any workload that has a sequential read component, including pure sequential read workloads.



AMP dramatically improves performance for common sequential and batch processing workloads. It also provides excellent performance synergy with DB2 by preventing table scans from being I/O bound and improves performance of index scans and DB2 utilities such as Copy and Recover. Furthermore, AMP reduces the potential for array hot spots, which result from extreme sequential workload demands<sup>6</sup>.

### 2.10.3 Intelligent Write Caching

With code level 5.42.xx.xx, an additional cache algorithm, referred to as the Intelligent Write Caching (IWC), has been implemented. IWC improves performance through better write cache management and a better destage order of writes. This algorithm is a combination of the known CLOCK, a predominantly read-cache algorithm, and CSCAN, an efficient write-cache algorithm. Out of this combination, IBM produced a powerful and widely applicable write-cache algorithm.

The CLOCK algorithm uses *temporal* ordering. It keeps a circular list of pages in memory, with the “hand” pointing to the oldest page in the list. When a page must be inserted in the cache, an *R* (recency) bit is inspected at the hand's location. If *R* is zero, the new page is put in place of the page that the hand points to and *R* is set to 1; otherwise, the *R* bit is cleared and set to zero. Then, the clock hand moves one step clockwise forward and the process is repeated until a page is replaced.

The CSCAN algorithm uses *spatial* ordering. The CSCAN algorithm is the circular variation of the SCAN algorithm. The SCAN algorithm tries to minimize the disk head movement when servicing read and write requests. It maintains a sorted list of pending requests along with the position on the drive of the request. Requests are processed in the current direction of the disk head, until it reaches the edge of the disk. At that point the direction changes. In the CSCAN algorithm, the requests are always served in the same direction. When the head has arrived at the outer edge of the disk, it returns to the beginning of the disk and services the new requests in this one direction only. This technique results in more equal performance for all head positions.

The basic idea of the new ICW is to maintain a sorted list of write groups, as in the CSCAN algorithm. The smallest and the highest write groups are joined, forming a circular queue. The additional new idea is to maintain a recency bit for each write group, as in the CLOCK algorithm. A write group is always inserted in its correct sorted position and the recency bit is set to zero at the beginning. When a write-hit occurs, the recency bit is set to one. The destage operation proceeds, where a destage pointer is maintained that scans the circular list looking for destage victims. Now, this algorithm only allows destaging of write groups whose recency bit is zero. The write groups with a recency bit of one are skipped and the recent bit is then turned off and reset to zero, which gives an extra life to those write groups that have been hit since the last time the destage pointer visited them.

Figure 2-31 on page 68 shows how this mechanism works.

In the DS8000 implementation, an IWC list is maintained for each rank. The dynamically adapted size of each IWC list is based on workload intensity on each rank. The rate of destage is proportional to the portion of NVS occupied by an IWC list (the NVS is shared across all ranks in a cluster). Furthermore, destages are smoothed out so that write-bursts are not translated into destage bursts.

---

<sup>6</sup> For a detailed description of AMP and the theoretical analysis for its optimal usage, see *AMP: Adaptive Multi-stream Prefetching in a Shared Cache* by Binny Gill and Luis Angel D. Bathen, USENIX File and Storage Technologies (FAST), February 13-16, 2007, San Jose, CA. For a more detailed description, see *Optimal Multistream Sequential Prefetching in a Shared Cache*, by Binny Gill and Luis Angel D. Bathen, in the ACM Journal of Transactions on Storage, October 2007.

## How Intelligent Write Caching Works

- Organize write groups in a sorted order forming a clock (temporal)
- Clock hand moves clockwise destaging write groups in order (spatial)
- Write groups are created with bit initialized to 0
- Clock hand can only destage groups with bit 0
- When clock hand encounters a write group with bit 1, it resets bit to zero, and skips it
- On a write to an existing group (hit), set the bit to 1

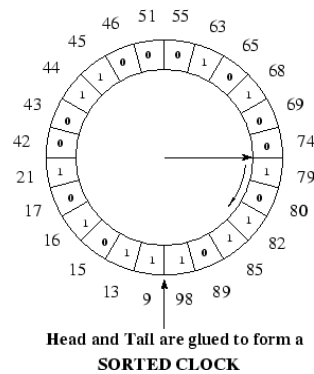


Figure 2-31 Intelligent Write Caching

In summary, IWC has better or comparable peak throughput to the best of CSCAN and CLOCK across a wide range of write cache sizes and workload configurations. In addition, even at lower throughputs, IWC has lower average response times than CSCAN and CLOCK.

## 2.11 Common performance considerations for System z

In this section, we discuss several topics that are specific to System z regarding the performance potential of the DS8000. We also discuss considerations for configuring and sizing a DS8000 that replaces older storage hardware in System z environments.

### 2.11.1 Host connections to System z servers

Figure 2-32 on page 69 partially shows a configuration where a DS8000 connects to FICON hosts. Note that this figure indicates only the connectivity to the Fibre Channel switched-disk subsystem through its I/O enclosure, symbolized by the rectangles.

Each I/O enclosure can hold up to four host adapters (HAs). The example in Figure 2-32 on page 69 shows only eight FICON channels connected to the first two I/O enclosures. Not shown is a second FICON director, which connects in the same fashion to the remaining two I/O enclosures to provide a total of 16 FICON channels in this particular example. The DS8100 disk storage subsystem provides up to 64 FICON channel ports. Again, note the efficient FICON implementation in the DS8000 FICON ports.

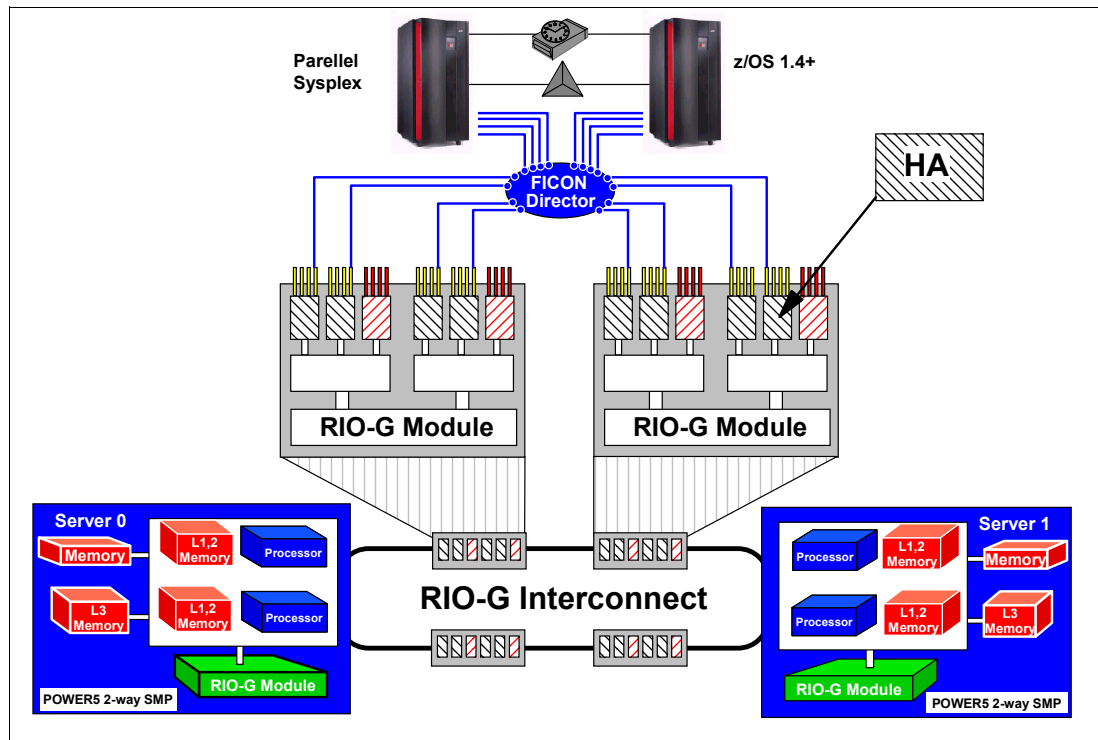


Figure 2-32 DS8100 front-end connectivity example, partial view

## High Performance FICON for System z (zHPF)

Enhancements have been made to z10 system architecture and the FICON protocol.

IBM introduced High Performance FICON, known as zHPF, with the z10 processor, and it is further enhanced by the new channels in the IBM zEnterprise System (zHPF requires certain control unit and z/OS levels). zHPF is a channel I/O architecture that replaces Modified Indirect Data Address Words (MIDAWs). Just as MIDAWs improved the efficiency of the channel subsystem, zHPF goes even further towards greatly improved channel efficiency. The zHPF architecture encompasses new protocols in the communication between the channel and the control unit, and a new channel program that is implemented by media manager. zHPF is transparent to DB2, just as MIDAWs were. At the present time, not all media manager I/Os can use zHPF. I/Os that access discontinuous pages are ineligible and format-writes are ineligible. On the z10, I/Os that read or write more than 64K are ineligible, but this restriction is removed on the zEnterprise System.

These enhancements optimize I/O for online transaction processing workloads. When exploited by the FICON channel, z/OS and the DS8000 control unit, zHPF can help reduce overhead and improve performance in the following ways:

- ▶ The maximum number of I/Os can be improved by up to 100% for small data transfers that are able to exploit zHPF.
- ▶ Production workloads with a mix of data transfer sizes can see up to 30 - 70% of FICON I/Os using zHPF, saving 10 - 30% channel utilization.
- ▶ Sequential I/O transferring less than a single track size (48 KB per I/O) can also benefit.
- ▶ Data accessed by DB2, PDSE, VSAM, and extended format SAM can benefit from zHPF and the new channel programs built by Media Manager.

The requirements are as follows:

- ▶ Only available on System z10®, on FICON Express2 and Express4
- ▶ z/OS V1R8 and later (or V1R7 with life cycle extension 5637-A01)
- ▶ IBM DS8000 Release 4.1 (LMC level 5.4.1.xx, bundle version 64.1.x.x] or later.
- ▶ Priced license feature with a monthly maintenance charge

Support for zHPF is an optional feature of the DS8000 series Turbo Models (931, 932, and 9B2) and is available with the High Performance FICON licensed feature indicator.

**Note:** Only extended count key data (ECKD) operations are eligible for zHPF. CKD operations continue to use CCWs.

zHPF together with FICON Channel, z/OS, and DS8000 control unit can help reduce overhead in I/O numbers. See Figure 2-33 for details.

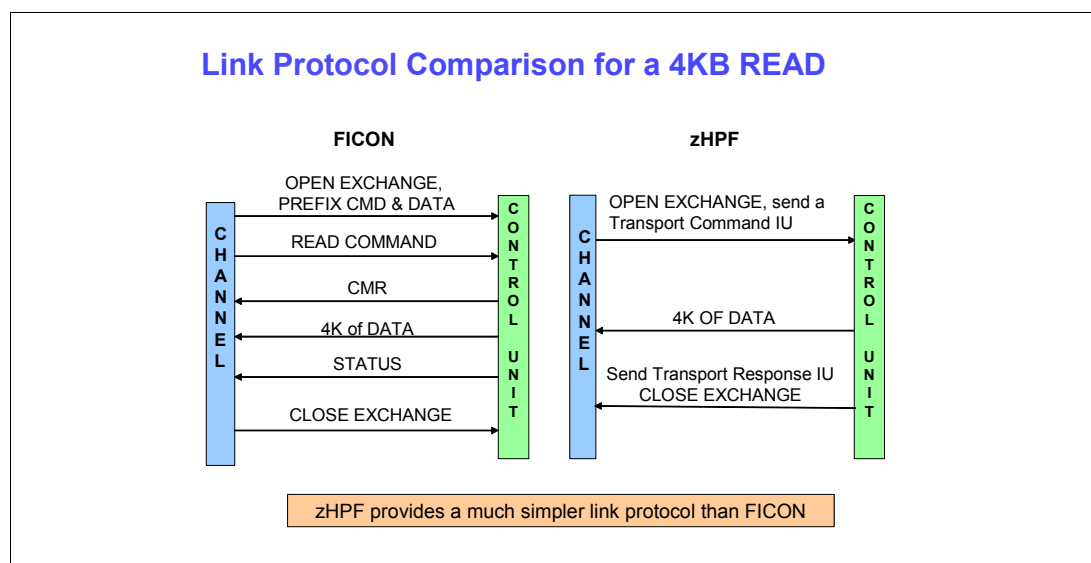


Figure 2-33 zHPF Link Protocol Comparison for a 4KB READ

## Modified Indirect Data Address Words (MIDAWs)

MIDAWs are a channel programming capability introduced with the IBM System z9® processor. The MIDAW facility is an extension to the pre-existing Indirect Data Address Word (IDAW) channel programming capability that has existed since the days of the IBM S/360 operating system, providing support for more efficient FICON channel programs.

MIDAWs are a new method of gathering data into and scattering data from discontiguous storage locations during an I/O operation. MIDAWs require the IBM System z9 server and IBM z/OS 1.7 (or APAR OA10984 with Release 1.6). MIDAWs are implemented by the Media Manager (M/M). To take advantage of M/M, users must use data set types, such as linear data sets and extended format data sets that enable the system to use M/M. The tuning aspects of the MIDAW facility are basic: Make sure that the processor and operating system are at the correct level, and use data set types that enable the system to use M/M.

The use of MIDAWs does not cause the bits to move any faster across the FICON link, but they reduce the number of frames and sequences flowing across the link, making the channel more efficient.

Channel efficiency is essentially the relationship of throughput to channel utilization. For a homogenous I/O stream of some uniform set of I/Os, the I/O rate, throughput, and channel utilization are all linearly related to each other. A channel becomes more efficient if the throughput at a particular level of utilization is increased.

Because the most significant performance benefit of MIDAWs is achieved with extended format data sets, we want to review the purpose of EF data sets. We review Media Manager and its role in the z/OS operating system. Then, we explore technical aspects of z/OS channel programming that can help you gain insight into what the MIDAW facility actually is, and why it improves the performance.

## Extended format (EF) data sets

In 1993, IBM introduced EF data sets. Both VSAM and non-VSAM (DSORG=PS) can be EF. In the case of non-VSAM data sets, a 32-byte suffix is appended to the end of every physical record (block) on DASD. VSAM appends the suffix to the end of every control interval (CI), which normally corresponds to a physical record. A 32 KB CI is split into two records to span tracks. This suffix is used to improve data reliability and facilitates other functions described below. Thus, for example, if the DCB BLKSIZE or VSAM CI size is equal to 8192, the actual block on DASD consists of 8224 bytes. The control unit itself does not distinguish between suffixes and user data. The suffix is transparent to the access method or database.

Besides reliability, EF data sets enable three functions: DFSMS striping, access method compression, and extended addressability (EA). DFSMS EA is especially useful for creating large DB2 partitions (DSSIZE greater than 4 GB). Striping can be used to increase sequential throughput, or to spread random I/Os across multiple logical volumes. DFSMS striping is especially useful for using multiple channels in parallel for one data set. The DB2 logs are often striped to optimize the performance of DB2 sequential inserts.

## z/Architecture channel programs

To understand what MIDAWs are, and why they are important to FICON performance, it is helpful to review the channel programming architecture of zSeries®, which mostly dates back to the days of S/360.

An I/O operation is represented by a *channel program*. A channel program consists of a series of (channel command words (CCWs) that form a chain. Media Manager uses Format 1 CCWs as illustrated in Figure 2-34.

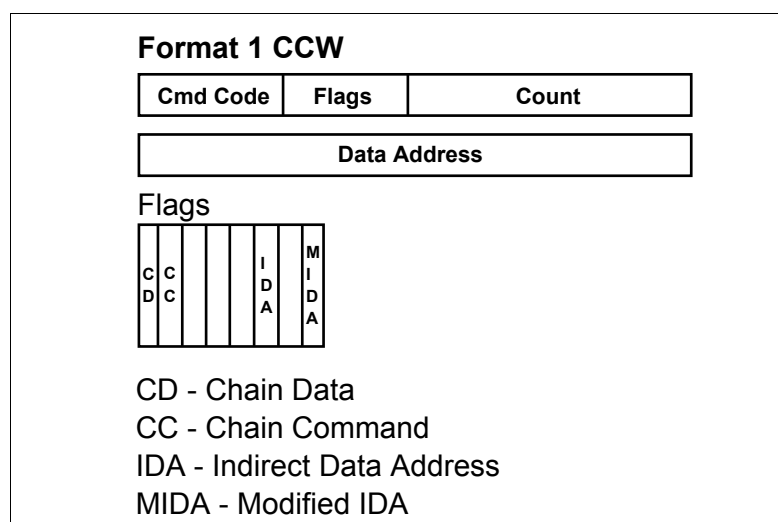


Figure 2-34 Channel command word

The command code specifies to the channel subsystem and the I/O device the operation to be executed; the count field specifies the number of bytes to transfer. When the channel subsystem has completed the transfer of information specified by a CCW, it can fetch the next CCW. Such fetching is called *chaining*, and the CCWs belonging to such a sequence are said to be chained. Two types of chaining are provided: chaining of data and chaining of commands. One flag in the CCW indicates data chaining and one indicates command chaining. The last CCW of a channel program is one where both chaining flags are off.

Unless the IDA flag is set, the data address in the CCW points directly at a continuous segment of real storage. Because ranges of virtual addresses may span discontinuous real 4 KB frames, direct addressing cannot generally be used to address more than 4 KB.

## Indirect data addressing (IDA)

IDA permits a single CCW to control the transfer of data that spans non-contiguous 4 KB frames in main storage. When the IDA flag is set, the data address in the CCW points to a list of words (IDAWs), each of which contains an address designating a data area within real storage. See Figure 2-35.

Prior to MIDAWs, Media Manager used Format 2 IDAWs, which enabled it to use data addresses above 2 GB (real addresses). Format 2 IDAWs designate 4 KB chunks, one IDAW for every 4 KB. The number of IDAWs is determined by however many IDAWs are needed to satisfy the count field in the CCW.

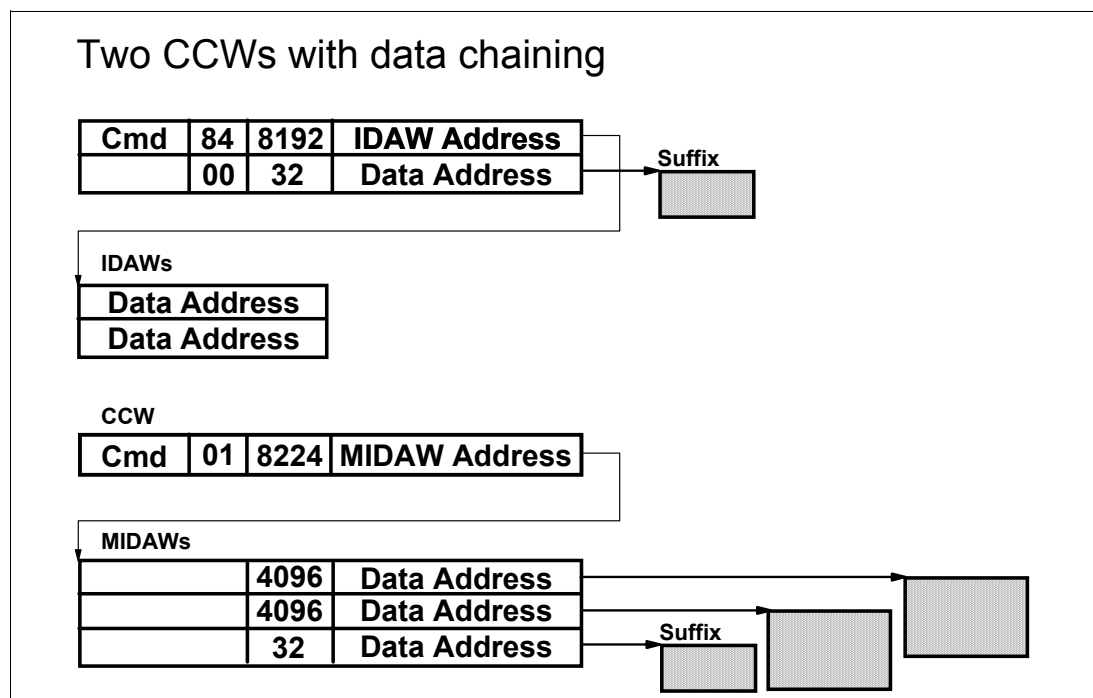


Figure 2-35 Transferring an 8 KB record from and to an EF data set

The first IDAW can designate any location, but all subsequent IDAWs must point at 4 KB boundaries. (Eliminating this restriction is an essential feature of MIDAWs.)

The MIDAW function provides the change of protocol for gathering data into and scattering data from discontinuous storage locations during an I/O operation. It improves FICON performance, especially when accessing DB2 databases, by reducing channel utilization and increasing throughput for parallel access streams.

Note that there is no change to ESCON, FICON, or control unit implementations. However, there is a greater chance that more *information units* (up to 16) can be sent to FICON control units in a single burst.

For detail information about how MIDAW works, the benefits, and why MIDAW is a requirement in the channel area, see *How does the MIDAW Facility Improve the Performance of FICON Channels Using DB2 and other workloads*, REDP-4201.

### 2.11.2 DS8000 size compared to older storage subsystems

We now look at sizing guidelines when the DS8000 replaces or take over the workload of an older model disk subsystem.

#### Guidelines

A sizing approach to follow is to propose how many ESS 800s might be consolidated into a DS8000 series model. From there, you can derive the number of ESS 750s, ESS F20s, and ESS E20s that can be consolidated into a DS8000. The older ESS models have a known relationship to the ESS 800. Further considerations are, for example, the connection technology used, such as ESCON, FICON, or FICON Express channels, and the number of channels.

In general, a properly configured DS8100 has the potential to provide the same or better numbers than two ESS 800s. Because the ESS 800 has the performance capabilities of two ESS F20s, a properly configured DS8100 can replace four ESS F20s. As the DS8000 series scales linearly, a well configured DS8300 has the potential to have the same or better numbers concerning I/O rates, sequential bandwidth, and response time than two DS8100s or four ESS 800s. Because the ESS 800 has roughly the performance potential of two ESS F20s, a corresponding number of ESS F20s can be consolidated. This approach applies also to the ESS 750, which has a similar performance behavior to that of an ESS F20.

When comparing the DS8000 series Turbo models 931, 932, and 9B2 to the predecessor DS8000 series models 921, 922, and 9A2, consider that the Turbo models feature the IBM POWER5™ processor. This processor can enable up to a 15% performance improvement in I/O operations per second in transaction processing workload environments compared to the IBM POWER5™ processor of the 921, 922, and 9A2 models.

#### Best approach is to use Disk Magic

The descriptions in the previous section are general guidelines. Still, the best approach is to use your installation workload RMF data as input to the Disk Magic<sup>7</sup> modelling tool. With your workload data and current configuration of disk subsystem units, Disk Magic can establish a base model from where it can project the necessary DS8000 units and their configuration that can absorb the present workload and also any future growth that you assume.

### 2.11.3 DS8000 processor memory size

Cache, or *processor memory* as the DS8000 term, is especially important to System z-based I/O. Processor memory or cache in the DS8000 contributes to high I/O rates and helps to minimize I/O response time.

Processor memory or cache can grow to up to 256 GB in the DS8300 and to 128 GB for the DS8100. This processor memory is subdivided into a data-in-cache portion, which holds data

<sup>7</sup> IBM has licensed Disk Magic from IntelliMagic since 1994 and used it to analyze and predict the performance of workloads on disk subsystems and technologies. In 2006, IntelliMagic released a user version of Disk Magic that supports disk storage system configurations from EMC, Hitachi Data Systems and HP in addition to IBM.

in volatile memory, and a persistent part of the memory, which functions as nonvolatile storage (NVS) to hold DASD fast-write (DFW) data until destaged to disk.

It is not only the pure cache size that accounts for good performance figures. Economical use of cache, such as 4 KB cache segments and smart, adaptive caching algorithms, are also as important to guarantee outstanding performance. These features are implemented in the DS8000 series.

Besides the potential for sharing data on a DS8000 processor memory level, cache is the main contributor to good I/O performance.

Use your current configuration and workload data and consider the following guidelines:

- ▶ Choose a cache size for the DS8000 series, which has a similar ratio between cache size and disk storage capacity to that of the currently used configuration.
- ▶ When you consolidate multiple disk storage units, configure the DS8000 processor memory or cache size as the sum of all cache from your current disk storage units.

For example, consider a DS8100 replacing four ESS F20s with 3.2 TB and 16 GB cache each. The ratio between cache size and disk storage for the ESS F20s is 0.5% with 16 GB/3.2 TB. The new DS8100 is configured with 18 TB to consolidate 4 x 3.2 TB plus extra capacity for growth. This configuration requires 90 GB of cache to keep the original cache-to-disk storage ratio. Round up to the next available memory size, which is 128 GB for this DS8100 configuration.

This ratio of 0.5% cache to backstore is considered high performance for z/OS environments. Standard performance suggests a ratio of 0.2% cache to backstore.

## 2.11.4 Channels

When sizing the DS8000, the number of channels you currently have in your configuration is also a matter to consider. With the trend to use 4 Gbps FICON channels on the System and in the DS8000, the number of channels can be reduced in most cases. In many cases four FICON channels is enough. For detailed planning, analyze the current workload.

## 2.11.5 Ranks and extent pool configuration

We already discussed ranks and extent pool configurations in 2.9.6, “Extent pools” on page 51. In this section, we discuss several specific System z topics, including SMS storage pools.

Note the independence of logical subsystems (LSSs) from ranks in the DS8000. Because an LSS is congruent with a System z logical control unit (LCU), we need to understand the implications. Having volumes within the same LCU, which is the same LSS, is now possible, but these volumes might reside in different ranks.

A horizontal pooling approach assumes that volumes within a logical pool of volumes, such as all DB2 volumes, are evenly spread across all ranks. This approach is independent of how these volumes are represented in LCUs. This section assumes horizontal volume pooling across ranks, which might be congruent with LCUs when mapping ranks accordingly to LSSs.

### Configure one extent pool for each single rank

When defining an extent pool, an affinity is created between a specific extent pool and a server.



Because of the virtualization of the Fibre Channel switched-disk subsystem, you might consider creating as many extent pools as there are RAID ranks in the DS8000. This approach can then work similarly to what is currently in the ESS. With this approach, you can control the placement of each single volume and where it ends up in the disk subsystem.

In the example in Figure 2-36, each rank is in its own extent pool. The evenly numbered extent pools have an affinity to the left server, server 0. The odd number extent pools have an affinity to the right server, server 1. When a rank is subdivided into extents, it gets assigned to its own extent pool.

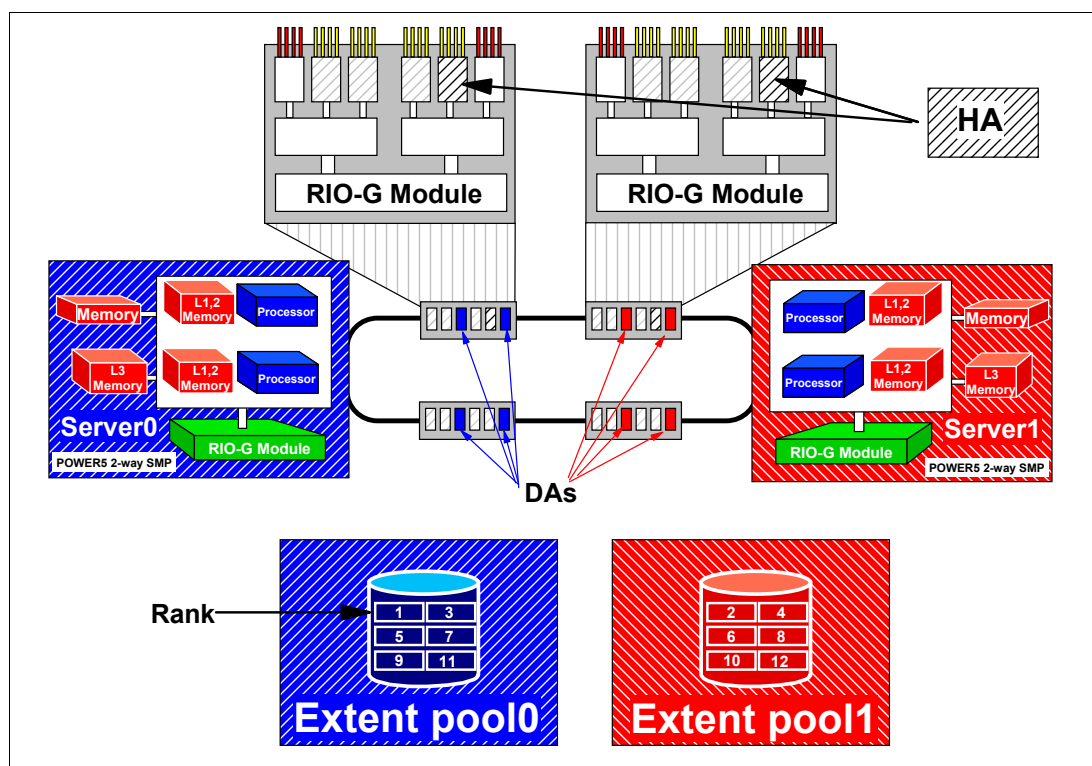


Figure 2-36 Extent pool affinity to processor complex with one extent pool for each rank

Now, all volumes that consist of extents from an extent pool have also a respective server affinity when scheduling I/Os to these volumes.

This approach allows you to place certain volumes in specific ranks to avoid potential clustering of many high activity volumes within the same rank. You can create system-managed storage (SMS) groups, which are congruent to these extent pools, to ease the management effort of such a configuration. However, you can still assign multiple storage groups when you are not concerned about the placement of less active volumes.

There is no affinity or certain preference between HA and processor complexes or servers in the DS8000. In this example, either one of the two HAs can address any volume in any of the ranks, which range here from rank number 1 - 12. Note, there is an affinity of DAS to the processor complex. A DA pair connects to a pair of switches. The first DA of this DA pair connects to the left processor complex or server 0. The second DA of this DA pair connects to the other processor complex or server 1.

## Minimize the number of extent pools

The other extreme is to create only two extent pools when the DS8000 is configured as count key data (CKD) storage only. You then subdivide the disk subsystem evenly between both processor complexes or servers as Figure 2-37 shows.

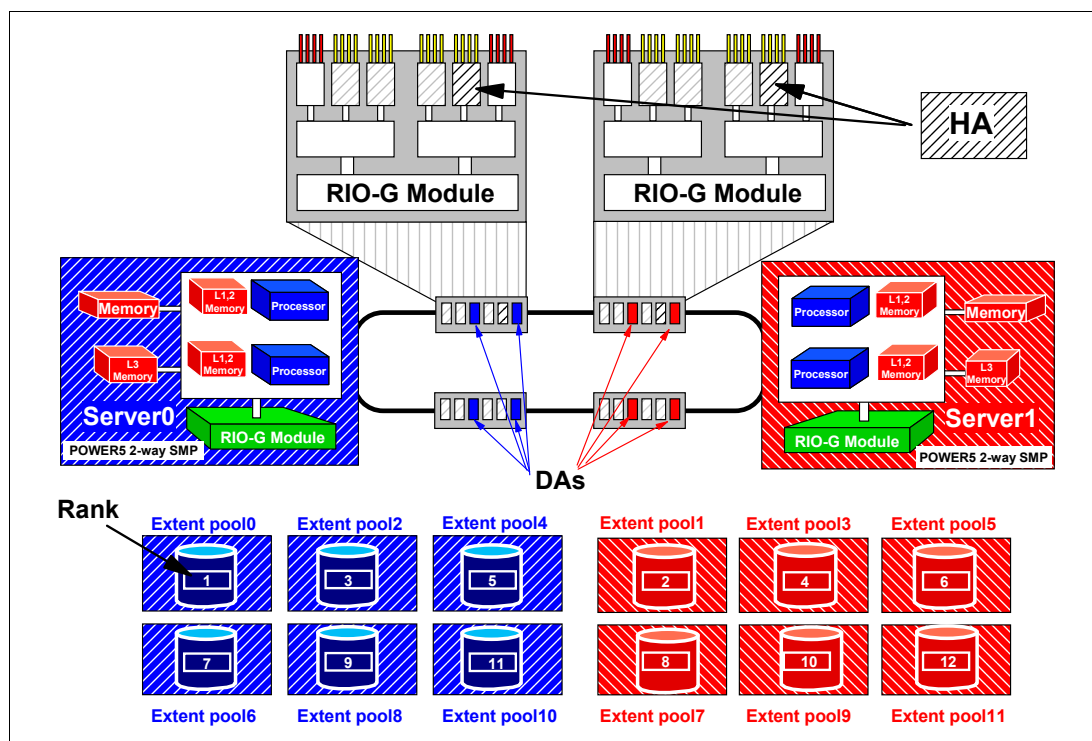


Figure 2-37 Extent pool affinity to processor complex with pooled ranks in two extent pools

Again, what is obvious here is the affinity between all volumes residing in extent pool 0 to the left processor complex, server 0, and the opposite is true for the volumes residing in extent pool 1 and their affinity to the right processor complex or server 1.

## Storage Pool Striping

The best way to configure a DS8000 is to set up multi-rank extent pools with about four to eight ranks per extent pool, to balance the ranks and extent pools between the two DS8000 servers. Also, use Storage Pool Striping.

## Plan for a reasonable number of extent pools

Figure 2-38 on page 77 presents a grouping of ranks into extent pools, which follows a similar pattern and discussion for grouping volumes or volume pools into SMS storage groups.

Create two general extent pools for all the average workload and the majority of the volumes, and subdivide these pools evenly between both processor complexes or servers. These pools contain the majority of the installed ranks in the DS8000. Then, consider two or four smaller extent pools with dedicated ranks for high performance workloads and their respective volumes. Consider defining storage groups, accordingly, that are congruent to the smaller extent pools.

If your DS8000 has disk drives of different speed or capacity, use separate extent pools for the drive types.

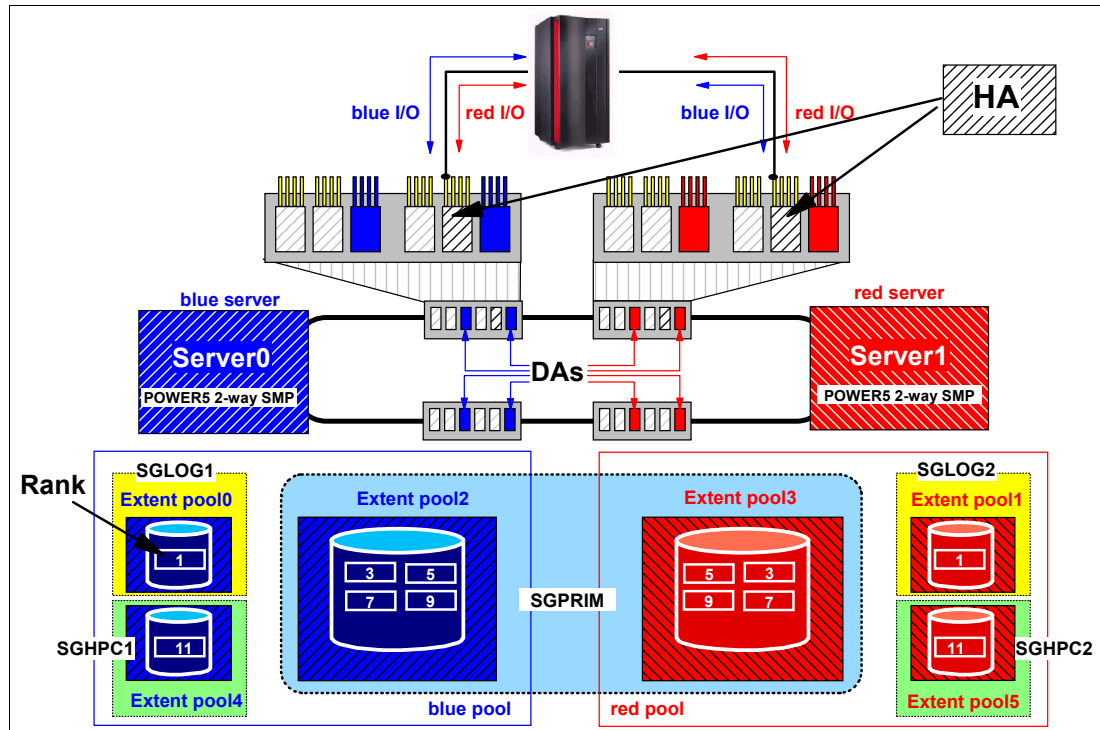


Figure 2-38 Mix of extent pools

Consider grouping the two larger extent pools into a single SMS storage group. SMS will eventually spread the workload evenly across both extent pools. This grouping allows a system-managed approach to place data sets automatically in the correct extent pools.

With more than one DS8000, consider configuring each DS8000 in a uniform fashion. A good approach is to group all volumes from all the large extent pools into one large SMS storage group, SGPRIM. Cover the smaller, high performance extent pools through discrete SMS storage groups for each DS8000.

With two of the configurations displayed in Figure 2-38, this configuration ends up with one storage group, SGPRIM, and six smaller storage groups. SGLOG1 contains extent pool 0 in the first DS8100 and the same extent pool in the second DS8100. Similar considerations are true for SGLOG2. For example, in a dual logging database environment, this technique allows you to assign SGLOG1 to the first logging volume and SGLOG2 for the second logging volume. For demanding I/O rates, and to satisfy a small set of volumes, consider keeping extent pool 4 and extent pool 5 in both DS8100s separate, through four distinct storage groups, SGHPC1-4.

Figure 2-38 shows, again, that there is no affinity between HA and processor complex or server. Each I/O enclosure connects to either processor complex. But there is an affinity between extent pool and processor complex and, therefore, an affinity between volumes and processor complex. This situation requires attention, as outlined previously, when you define your volumes.

## 2.12 Copy Services

Copy Services is a collection of functions that provide disaster recovery, data migration, and data duplication functions. With the Copy Services functions, for example, you can create backup data with little or no disruption to your application, and you can back up your application data to the remote site for the disaster recovery.

In addition to using the DS GUI, the DS CLI, or TPC for Replication, System z users have the following interfaces available for Copy Services management:

- ▶ TSO commands
- ▶ ICKDSF utility commands
- ▶ DFSMSdss utility
- ▶ ANTRQST application programming interface

We differentiate copy services between the ESS and DS8000 storage subsystem. The DS8000 storage subsystem has more functionality than the ESS storage subsystem. This section describes the following copy services:

- ▶ Copy Services with ESS
- ▶ Copy Services with DS8000

### 2.12.1 Copy Services with ESS

DFSMS provides Advanced Copy Services that include a hardware and software solution to help you manage and protect your data. These solutions help ensure that your data remains available 24 hours a day, seven days a week. Advanced Copy Services provide solutions to disaster recovery, data migration, and data duplication. Many of these functions run on the IBM TotalStorage Enterprise Storage Server (ESS). See Figure 2-39.

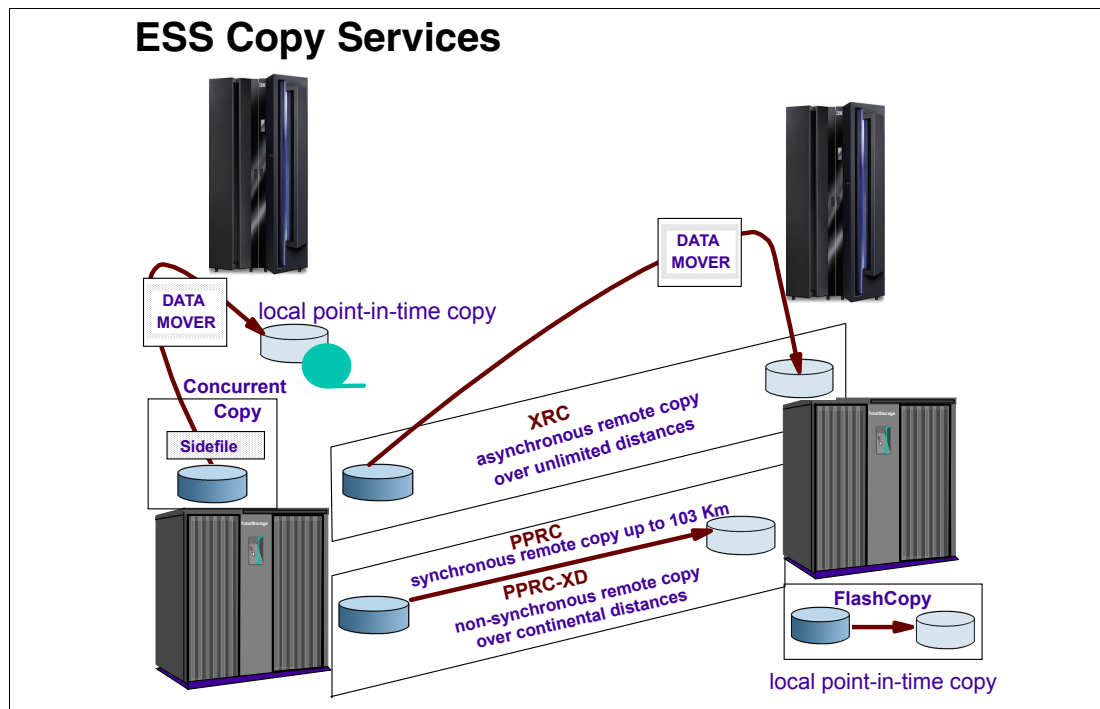


Figure 2-39 ESS Copy Services

With DFSMS, you can perform the following data management functions:

- ▶ Use remote copy function to prepare for disaster recovery.
- ▶ Move your PPRC data more easily.

Remote copy function provides two options that enable you to maintain a current copy of your data at a remote site. These two options are used for disaster recovery and workload migration:

- ▶ Peer-to-peer remote copy (PPRC)
- ▶ Extended remote copy (XRC)

Two types of copy functions are available:

- ▶ An instantaneous copy where all the late updates in the primary are not copied. It is used for fast backups and data replication in general. The examples in ESS are Concurrent Copy and Flash Copy.
- ▶ Mirroring, a never-ending copy where all the updates are mirrored as fast as possible. It is used for disaster recovery and planned outages. The example in ESS are Enhanced PPRC service and XRC.

### Peer-to-peer remote copy (PPRC)

PPRC is a hardware solution that provides rapid and accurate disaster recovery, and a solution to workload movement and device migration. Updates made on the primary DASD volumes are synchronously shadowed to the secondary DASD volumes. The local storage subsystem and the remote storage subsystem are connected through a communications link called a *PPRC path*. You can use one of the following protocols to copy data using PPRC:

- ▶ ESCON
- ▶ Fibre Channel Protocol

**Note:** Fibre Channel Protocol is supported only on ESS Model 800 with the appropriate Licensed Internal Code level and the PPRC Version 2 feature enabled.

PPRC provides a synchronous volume copy across ESS controllers. The copy is done from one controller (the one having the primary logical device) to the other (having the secondary logical device). It is synchronous because the task that is doing the I/O waits for the guarantee that the copy was executed. With ESCON channels, a performance penalty exists for distances longer than 10 km. PPRC is used for disaster recovery, device migration, and workload migration; for example, it enables you to switch to a recovery system in the event of a disaster in an application system.

You can issue the CQUERY command to query the status of one volume of a PPRC volume-pair or to collect information about a volume in the simplex state. The CQUERY command is modified and enabled to report on the status of S/390-attached CKD devices.

See *z/OS DFSMS Advanced Copy Services*, SC35-0428, for further information about the PPRC service and the CQUERY command.

### Peer-to-peer remote copy extended distance (PPRC-XD)

When you enable the PPRC-XD feature, the primary and recovery storage control sites can be separated by long distances. Updates made to a PPRC primary volume are sent to a secondary volume asynchronously, thus requiring less bandwidth.

If you are trying to decide whether to use synchronous or asynchronous PPRC, consider the differences between the two modes:

- ▶ When you use synchronous PPRC, no data loss occurs between the last update at the primary system and the recovery site, but it increases the impact to applications and uses more resources for copying data.
- ▶ Asynchronous PPRC using the extended distance feature reduces impact to applications that write to primary volumes and uses less resources for copying data, but data might be lost if a disaster occurs. To use PPRC-XD as a disaster recovery solution, customers need to periodically synchronize the recovery volumes with the primary site and make backups to other DASD volumes or tapes.

PPRC-XD is a non-synchronous version of PPRC, which means that host updates to the source volume are not delayed by waiting for the update to be confirmed in the secondary volume. It also means that the sequence of updates on the secondary volume is not guaranteed to be the same as on the primary volume.

PPRC-XD is an excellent solution for the following tasks:

- ▶ Remote data copy
- ▶ Remote data migration
- ▶ Offsite backup
- ▶ Transmission of inactive database logs
- ▶ Application disaster recovery solutions based on periodic point-in-time (PIT) copies of the data, if the application tolerates short interruptions (application quiesce)

PPRC-XD can operate at very long distances (such as continental distances), well beyond the 103 km that is supported for PPRC synchronous transmissions, and with minimal impact on the application. The distance is limited only by the network and channel extender technology capabilities.

### **Extended remote copy (XRC)**

XRC combines hardware and software to provide continuous data availability in a disaster recovery or workload movement environment. XRC provides an asynchronous remote copy solution for both system-managed and non-system-managed data to a second, remote location.

XRC relies on the IBM TotalStorage Enterprise Storage Server, IBM 3990, RAMAC Storage Subsystems, and DFSMSdfp. The 9393 RAMAC Virtual Array (RVA) does not support XRC for source volume capability.

XRC relies on the system data mover, which is part of DFSMSdfp. The system data mover is a high-speed data movement program that efficiently and reliably moves large amounts of data between storage devices. XRC is a continuous copy operation, and it is capable of operating over long distances (with channel extenders). It runs unattended, without involvement from the application users. If an unrecoverable error occurs at your primary site, the only data that is lost is data that is in transit between the time when the primary system fails and the recovery at the recovery site.

You can implement XRC with one or two systems. Suppose that you have two systems: an application system at one location, and a recovery system at another. With these two systems in place, XRC can automatically update your data on the remote disk storage subsystem as you make changes to it on your application system. You can use the XRC suspend/resume service for planned outages. You can still use this standard XRC service on systems attached to the ESS if these systems are installed with the toleration or transparency support.

Coupled extended remote copy (CXRC) allows XRC sessions to be coupled together to guarantee that all volumes are consistent across all coupled XRC sessions. CXRC can manage thousands of volumes. IBM TotalStorage XRC Performance Monitor provides the ability to monitor and evaluate the performance of a running XRC configuration.

## Concurrent copy

Concurrent copy is an extended function that enables data center operations staff to generate a copy or a dump of data while applications are updating that data. Concurrent copy delivers a copy of the data, in a consistent form, as it existed before the updates took place.

## FlashCopy

FlashCopy is a point-in-time copy services function that can quickly copy data from a source location to a target location. FlashCopy enables you to make copies of a set of tracks, with the copies immediately available for read or write access. This set of tracks can consist of an entire volume, a data set, or just a selected set of tracks. The primary objective of FlashCopy is to create a copy of a source volume on the target volume. This copy is called a *point-in-time copy*. Access to the point-in-time copy of the data on the source volume is through reading the data from the target volume. The actual point-in-time data that is read from the target volume might or might not be physically stored on the target volume. The ESS FlashCopy service is compatible with the existing service provided by DFSMSdss. Therefore, you can invoke the FlashCopy service on the ESS with DFSMSdss.

The ESS has had two versions of FlashCopy:

### ► FlashCopy Version 1

The original implementation of FlashCopy in the ESS had the following characteristics:

- FlashCopy Version 1 worked at the volume level only.
- The source and target volumes needed to be in the same ESS logical subsystem (LSS).
- A source and a target volume could only be involved in one FlashCopy relationship at a time.

FlashCopy Version 2 is available as an optional feature of the ESS.

### ► FlashCopy Version 2

Contains new functionality; provides all the functions of FlashCopy Version 1, plus the following enhancements:

- In addition to volume copy, data-set FlashCopy is also supported.
- Multiple FlashCopy relationships can be present on a volume and extent level.
- When doing extent FlashCopy, target tracks do not have to be in the same location as source tracks. Target tracks can even be on the same volume as the source tracks.
- Source and target volumes can be on the same or different LSSs.
- Incremental copies of established FlashCopy relationships are supported.
- Consistency groups can be enabled when establishing FlashCopy relationships.

FlashCopy Version 2 is available as an optional feature of the ESS. Current FlashCopy Version 1 can upgrade to Version 2 installing ESS Licensed Internal Code 2.2.0 or later.

## 2.12.2 Copy Services with DS8000

The Copy Services functions run on the DS8000 storage unit and support open systems and System z environments. These functions are supported also in the DS6000 series and, with some limitation, on the previous generation of storage systems, the IBM TotalStorage Enterprise Storage Server (ESS) models.

Copy Services in the DS8000 includes the following optional licensed functions:

- ▶ IBM System Storage Flash Copy and IBM FlashCopy SE, which are point-in-time copy functions
- ▶ Remote mirror and copy functions, which include the following items:
  - IBM System Storage Metro Mirror, previously known as synchronous PPRC
  - IBM System Storage Global Copy, previously known as PPRC Extended Distance
  - IBM System Storage Global Mirror, previously known as asynchronous PPRC
  - IBM System Storage Metro/Global Mirror, a three-site solution to meet the most rigorous business resiliency needs
- ▶ Additionally for the System z users, the following items are available:
  - z/OS Global Mirror, previously known as Extended Remote Copy (XRC)
  - z/OS Metro/Global Mirror, a 3-site solution that combines z/OS Global Mirror and Metro Mirror (previously called PPRC)

Many design characteristics of the DS8000 and its data copy and mirror capabilities and features contribute to the protection of your data, 24 hours a day and seven days a week.

System z has following interfaces for managing Copy Services:

- ▶ TSO commands
- ▶ ICKDSF utility commands
- ▶ ANTRQST application programming interface (API)
- ▶ DFSMSdss utility

### ***DS8000 Copy Services matrix***

The matrix in Figure 2-40 on page 83 shows which copy service is possible.



## DS8000 Copy Services Matrix

Device Is May Become	GMz <sup>10</sup> (XRC) Pri	GMz <sup>10</sup> (XRC) Sec	MM or GC Pri	MM or GC Sec	GM Pri	GM Sec	FLC Source	FLC Target	Inc FLC Source	Inc FLC Target	SE FLC <sup>12</sup> Source	SE FLC <sup>12</sup> Target	Con Copy Source
GMz <sup>10</sup> (XRC) Pri	No	Yes <sup>11</sup>	Yes	No	No	No	Yes	No	No <sup>7</sup>	No <sup>7</sup>	Yes	Yes	Yes
GMz <sup>10</sup> (XRC) Sec	Yes <sup>11</sup>	No	Yes	No <sup>5</sup>	Yes	No <sup>5</sup>	Yes	No <sup>5</sup>	Yes	No <sup>5</sup>	Yes	No <sup>5</sup>	Yes
MM or GC Primary	Yes	Yes	No	Yes <sup>1</sup>	No <sup>6</sup>	Yes <sup>1</sup>	Yes	Yes	Yes	Yes <sup>2</sup>	Yes	Yes	Yes
MM or GC Secondary	No	No <sup>5</sup>	Yes <sup>1</sup>	No	Yes <sup>1</sup>	No	Yes	Yes <sup>8</sup>	Yes	Yes	Yes	Yes <sup>8</sup>	No
GM Pri	No	Yes	No <sup>6</sup>	Yes <sup>1</sup>	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GM Sec	No	No <sup>5</sup>	Yes <sup>1</sup>	No	No	No	Yes	Yes <sup>8</sup>	Yes <sup>9</sup>	No	Yes	Yes	No
FLC Source	Yes	Yes	Yes	Yes	Yes	Yes	Yes <sup>3,4</sup>	Yes <sup>4</sup>	Yes <sup>3</sup>	Yes <sup>13</sup>	Yes	Yes	Yes
FLC Target	No	No <sup>5</sup>	Yes <sup>2</sup>	No	No	Yes <sup>9</sup>	Yes <sup>4</sup>	Yes <sup>4</sup>	No	No	Yes	Yes	No
Inc FLC Source	No <sup>7</sup>	Yes	Yes	Yes	Yes	Yes <sup>9</sup>	Yes	No	No	Yes <sup>13</sup>	Yes	No	Yes
Inc FLC Target	No <sup>7</sup>	No <sup>5</sup>	Yes <sup>2</sup>	No	No	No	No	No	Yes	No	No	No	No
SE FLC <sup>12</sup> Source	Yes	Yes	Yes	Yes	Yes	Yes	Yes <sup>3,4</sup>	Yes <sup>4</sup>	Yes	No	Yes	Yes <sup>4</sup>	Yes
SE FLC <sup>12</sup> Target	No	No <sup>5</sup>	Yes <sup>2</sup>	No	No	Yes <sup>9</sup>	Yes <sup>4</sup>	Yes <sup>4</sup>	No	No	Yes	Yes	No
Con Copy Source	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Figure 2-40 DS8000 Copy Services Matrix

Numbered notes about the DS8000 Copy Services Matrix in Figure 2-40 are as follows:

- GMz (XRC) Pri is only in a Metro/Global Copy (supported on ESS) or a Metro/Global Mirror Environment (supported on ESS and DS8000).
- Note the following information for FlashCopy V2 at LIC 2.4.0 and higher on ESS800 (DS6000 and DS8000 utilize FlashCopy V2 by default):
  - You must specify the proper parameter to perform this task.
  - Metro Mirror primary goes from full duplex to copy pending until all of the flashed data is transmitted to remote:
    - Entire volume for a full volume FlashCopy
    - Delta from the last increment if Incremental FlashCopy
    - Entire data set if Data Set Level FlashCopy
  - Global Mirror primary cannot be a FlashCopy target. A FlashCopy Target can become a Global Mirror primary.
- FlashCopy V2 provides Multiple Relationships is included.
- FlashCopy V2 Data Set FlashCopy is supported with fully provisioned FlashCopy volumes on z/OS only.
- Although the Storage Controller does not enforce this restriction, it is not recommended.
- A volume may be converted between the states Global Mirror primary, Metro Mirror primary, and Global Copy primary by using commands, but the two relations cannot exist at the same time (such as multi-target).

7. Global Mirror for z/OS (XRC) Primary, Global Mirror Secondary, Incremental FlashCopy Source and Incremental FlashCopy Target all use the Change Recording Function. For a particular volume, only one of these relationships may exist.  
Incremental FlashCopy relationship from A to B can be changed and become a relationship in the reverse direction from B to A by issuing a Fast Reverse Restore (FRR) command after reversing the FlashCopy relationship to the target.
8. Updates to the affected extents result in the implicit removal of the FlashCopy relationship, if the relationship is not persistent.
9. This relationship must be the FlashCopy relationship associated with Global Mirror, which means there may not be a separate Incremental FlashCopy relationship. A Fast Reverse Restore (FRR) switches a GM secondary to an FLC target.
10. Global Mirror for z/OS (GMz) is supported on ESS and DS8000.
11. To ensure Data Consistency, the XRC Journal volumes must also be copied.
12. Space-efficient FlashCopy requires DS8000 R3 plus an appropriate SE FLC capacity license. SE FLC volumes cannot be dynamically expanded while in a copy services relationship. Data set level FLC is not currently supported for SE FLC volumes.

### **Interoperability**

Remote mirror and copy pairs can be established only between disk subsystems of the same (or similar) type and features. For example, a DS8000 can have a remote mirror pair with another DS8000, a DS6000, an ESS 800, or an ESS 750. It cannot have a remote mirror pair with an RVA or an ESS F20. Note that all disk subsystems must have the appropriate features installed. If your DS8000 is mirrored to an ESS disk subsystem, the ESS must have PPRC Version 2 (which supports Fibre Channel links) with the appropriate Licensed Internal Code level.

DS8000 interoperability information can be found in the IBM System Storage Interoperability Center (SSIC); start at the following web site:

<http://www.ibm.com/systems/support/storage/config/ssic>

**Note:** The DS8000 does not support ESCON links for remote mirror and copy operations. If you want to establish a remote mirror relationship between a DS8000 and an ESS 800, you have to use FCP links.

### **FlashCopy and IBM FlashCopy SE**

FlashCopy and FlashCopy SE provide point-in-time copy capability for logical volumes with minimal interruption to applications, and enable access to both the source and target copies immediately.

In this section, we discuss basic characteristics and options of FlashCopy and FlashCopy SE. For a more detailed discussion about these topics, see *DS8000 Copy Services for IBM System z*, SG24-6787.

FlashCopy creates a point-in-time copy of the data. When a FlashCopy operation is invoked, it takes only a few seconds to complete the process of establishing a FlashCopy source and target volume pair, and creating the necessary control bitmaps. Thereafter, you have access to a point-in-time copy of the source volume, as though all the data had been copied. As soon as the pair has been established, you can read and write to both the source and target volumes.

Two variations of FlashCopy are available:

- ▶ Standard FlashCopy uses a normal volume as target volume. This target volume has to have the same size (or larger) as the source volume and that space is allocated in the storage subsystem.
- ▶ FlashCopy SE uses track space-efficient volumes as FlashCopy target volumes. A track space-efficient target volume has a virtual size that is equal to or greater than the source volume size. However, space is not allocated for this volume at the time the volume is created and the FlashCopy initiated. Only when updates are made to the source volume are any original tracks of the source volume that are modified are copied to the track space-efficient volume. Space in the repository is allocated for just these tracks or for any write to the target itself.

**Note:** Both FlashCopy and FlashCopy SE can coexist in a DS8000.

After a FlashCopy relationship is established, and if it is a standard FlashCopy, you have the option to start a background process (to copy all the tracks from the source to the target volume) or specify the no-copy option to not automatically do the physical copy of the entire volume. With FlashCopy SE, the no-copy option is enforced<sup>8</sup>. See Figure 2-41 for an illustration of these FlashCopy basic concepts.

**Note:** In this section, *track* means a piece of data in the DS8000; the DS8000 uses the logical tracks to manage the Copy Services functions.

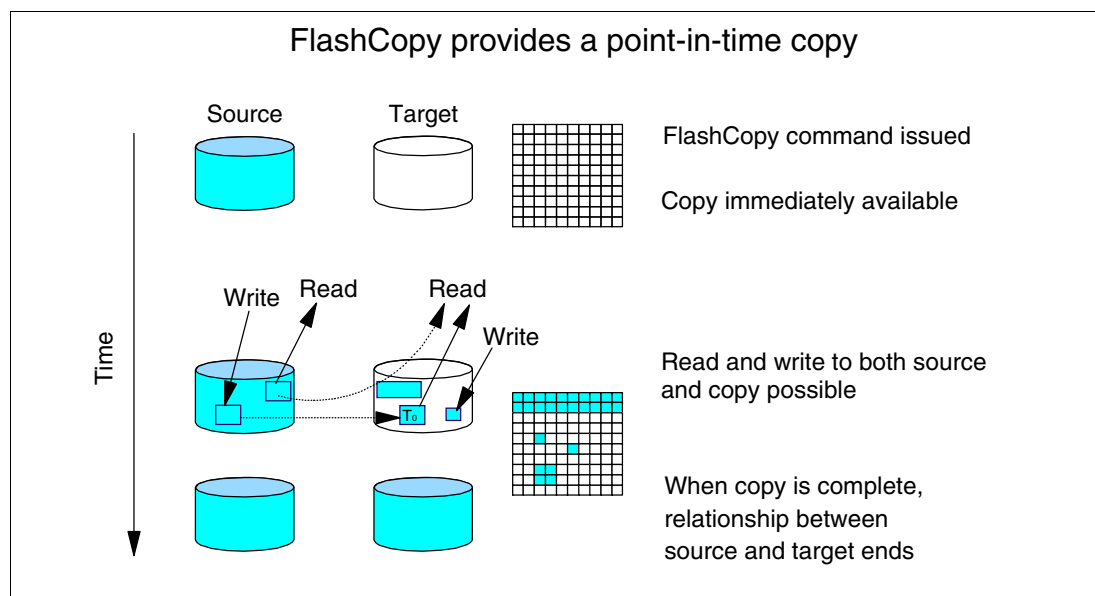


Figure 2-41 FlashCopy concept

<sup>8</sup> Because the intent of DB2 is to build a full background back-up copy, it uses the copy function. DB2 does not use FlashCopy SE volumes.

If you access the source or the target volumes during the background copy, standard FlashCopy manages these I/O requests as follows:

- Read from the source volume.

When a read request goes to the source volume, it is read from the source volume.

- Read from the target volume.

When a read request goes to the target volume, FlashCopy checks the bitmap and the following operations are performed:

- If the point-in-time data was already copied to the target volume, it is read from the target volume.
- If the point-in-time data has not been copied yet, it is read from the source volume.

- Write to the source volume.

When a write request goes to the source volume, first the data is written to the cache and persistent memory (write cache). Then, when the update is destaged to the source volume, FlashCopy checks the bitmap and the following operations are performed:

- If the point-in-time data was already copied to the target, then the update is written to the source volume.
- If the point-in-time data has not been copied yet to the target, then first it is copied to the target volume, and after that the update is written to the source volume.

- Write to the target volume.

Whenever data is written to the target volume while the FlashCopy relationship exists, the storage subsystem makes sure that the bitmap is updated. This way, the point-in-time data from the source volume never overwrites updates that were done directly to the target.

Although the background copy can slightly affect your applications' performance because the physical copy needs some storage resources, the impact is minimal because the host I/O has priority over the background copy. If you want, you can issue standard FlashCopy with the no background copy option.

If you invoke standard FlashCopy with the no background copy option or FlashCopy SE, the FlashCopy relationship is established without initiating a background copy. Therefore, you can minimize the impact of the background copy. When the DS8000 receives an update to a source track in a FlashCopy relationship, a copy of the point-in-time data is copied to the target volume so that it is available when the data from the target volume is accessed. This option is useful when you do not need to issue FlashCopy in the opposite direction.

The point-in-time copy that is created by FlashCopy is typically used where you need a copy of the production data produced with little or no application impact (depending on the application). The point-in-time copy created by FlashCopy can be used for online backup, testing new applications, or for creating a database for data mining purposes. The copy looks exactly like the original source volume and is an instantly available, binary copy.

IBM FlashCopy SE is designed for temporary copies. Because the target is smaller than the source, a background copy does not make much sense and is not permitted with IBM FlashCopy SE. FlashCopy SE is optimized for use cases in which about 5% of the source volume is updated during the life of the relationship. If more than 20% of the source is expected to change, standard FlashCopy is likely a better choice. Durations for typical use cases are expected to generally be less than eight hours unless the source data has little write activity. FlashCopy SE can also be used for copies that will be kept long-term if it is known that the FlashCopy relationship will experience few updates to the source and target volumes.

Standard FlashCopy generally has superior performance to FlashCopy SE. If performance on the source or target volumes is important, use standard FlashCopy.

The following scenarios are good choices for IBM FlashCopy SE:

- ▶ To create a temporary copy with IBM FlashCopy SE then back it up to tape
- ▶ Temporary point-in-time copies for application development or DR testing
- ▶ Online backup for different points in time, for example to protect your data against virus infection
- ▶ Checkpoints (only if the source volumes will undergo moderate updates)
- ▶ FlashCopy target volumes in a Global Mirror (GM) environment. If the repository space becomes completely full, writes to the GM target volume is inhibited and the GM session is suspended. Global Mirror is explained in “Global Mirror” on page 95.

## FlashCopy options

FlashCopy has many options and expanded functions for data copy. We explain several options and capabilities in this section:

- ▶ Incremental FlashCopy
- ▶ Data set level FlashCopy
- ▶ Multiple Relationship FlashCopy
- ▶ Consistency Group FlashCopy
- ▶ Establish FlashCopy on existing Metro Mirror or Global Copy primary
- ▶ Persistent FlashCopy
- ▶ Inband commands over remote mirror link
- ▶ Remote Pair FlashCopy

### ***Incremental FlashCopy***

Incremental FlashCopy (refresh target volume) provides the ability to *refresh* a logical unit number (LUN) or volume involved in a FlashCopy relationship. When a subsequent FlashCopy operation is initiated, only the tracks changed on both the source and target need to be copied from the source to the target. The direction of the *refresh* can also be reversed.

In many cases, at most 10 - 20% of the entire data is changed in a day. In this situation, if you use this function for daily backup, you can save the time for the physical copy of FlashCopy.

Figure 2-42 explains the basic characteristics of Incremental FlashCopy.

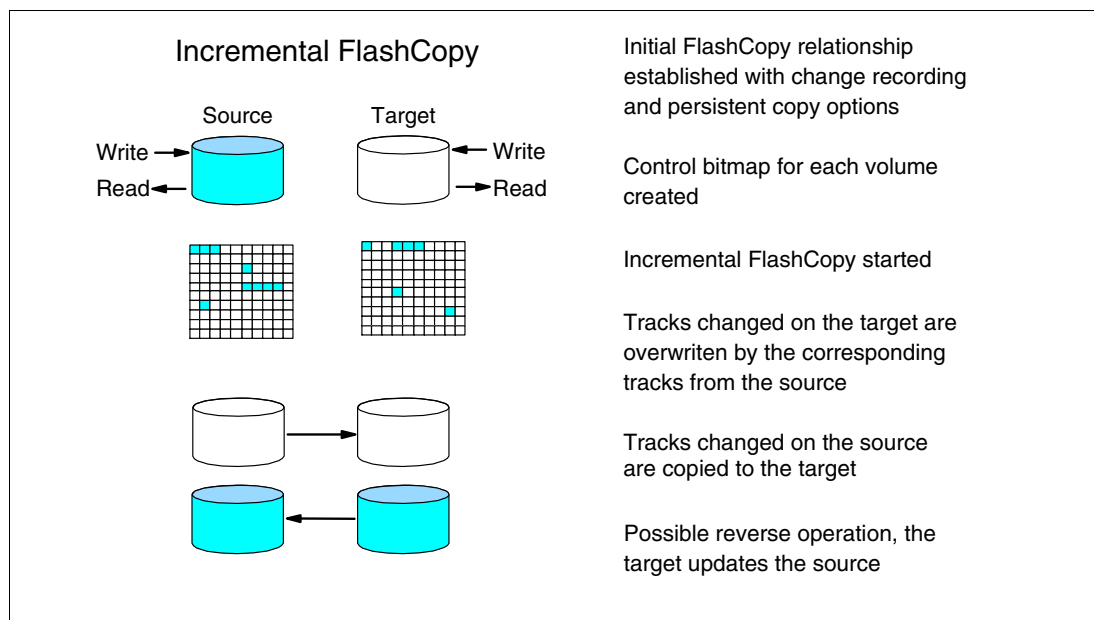


Figure 2-42 Incremental FlashCopy

By using Incremental FlashCopy option the following steps occur:

1. You issue full FlashCopy with the *change recording* option. This option is for creating change recording bitmaps in the storage unit. The change recording bitmaps are used for recording the tracks which are changed on the source and target volumes after the last FlashCopy.
2. After creating the change recording bitmaps, Copy Services records the information for the updated tracks to the bitmaps. The FlashCopy relationship persists even if all of the tracks have been copied from the source to the target.
3. The next time you issue Incremental FlashCopy, Copy Services checks the change recording bitmaps and copies only the changed tracks to the target volumes. If some tracks on the target volumes are updated, these tracks are overwritten by the corresponding tracks from the source volume.

You can also issue incremental FlashCopy from the target volume to the source volumes with the *reverse restore* option. The reverse restore operation cannot be done unless the background copy in the original direction has finished.

### **Data set FlashCopy**

Data set FlashCopy allows a FlashCopy of a data set in a System z environment. See Figure 2-43 on page 89.

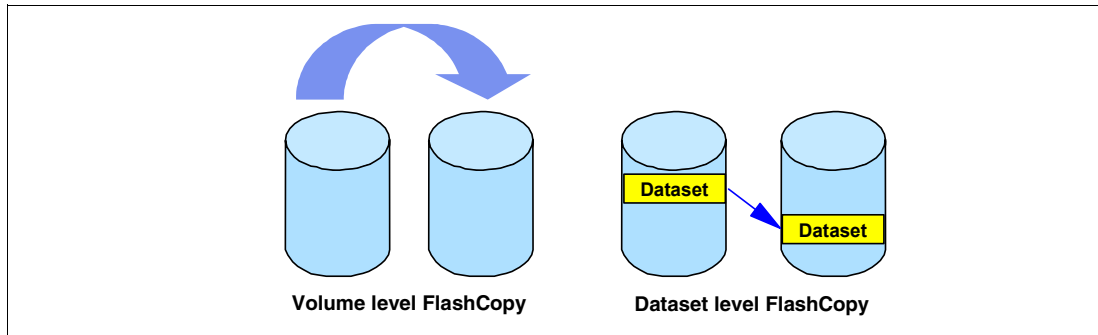


Figure 2-43 Data set FlashCopy

### Multiple Relationship FlashCopy

Multiple Relationship FlashCopy allows a source to have FlashCopy relationships with multiple targets simultaneously. A source volume or extent can be copied (FlashCopy) to up to 12 target volumes or target extents, as illustrated in Figure 2-44.

**Note:** If a FlashCopy source volume has more than one target, that source volume can be involved only in a single incremental FlashCopy relationship.

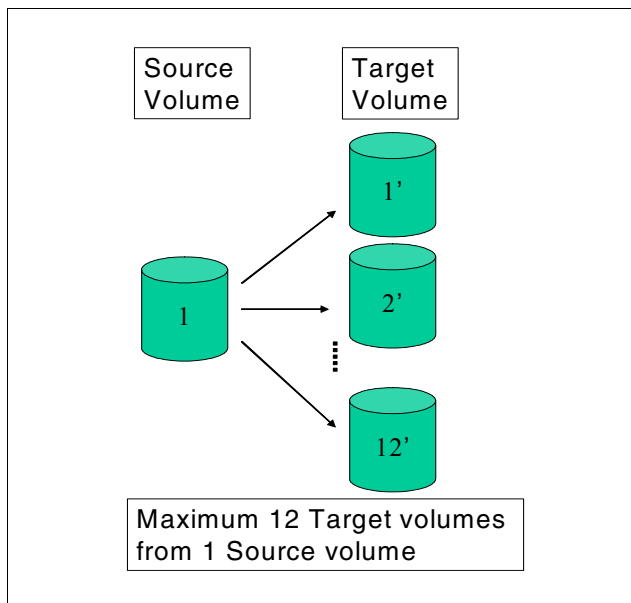


Figure 2-44 Multiple Relationship FlashCopy

### Consistency Group FlashCopy

Consistency Group FlashCopy allows you to freeze (temporarily queue) I/O activity to a LUN or volume. Consistency Group FlashCopy helps you to create a consistent point-in-time copy across multiple volumes, and even across multiple storage units.

If a consistent point-in-time copy across many logical volumes is required, and you do not want to quiesce host I/O or database operations, you can use Consistency Group FlashCopy to create a consistent copy across multiple logical volumes in multiple storage units.

To create this consistent copy, you issue a set of *establish* FlashCopy commands with a *freeze* option, which holds off host I/O to the source volumes. This means that Consistency Group

FlashCopy provides the capability to temporarily queue (at the host I/O level, not the application level) subsequent write operations to the source volumes that are part of the Consistency Group. During the temporary queuing, establishing FlashCopy is completed. The temporary queuing continues until this condition is reset by the **Consistency Group Created** command or the timeout value expires (the default is two minutes).

After all of the *establish* FlashCopy requests have completed, a set of Consistency Group Created commands must be issued using the same set of DS network interface servers. The Consistency Group Created commands are directed to each logical subsystem (LSS) involved in the Consistency Group. The Consistency Group Created command allows the write operations to resume to the source volumes.

This operation is illustrated in Figure 2-45.

For a more detailed discussion of the concept of *data consistency* and how to manage the Consistency Group operation, see *DS8000 Copy Services for IBM System z*, SG24-6787.

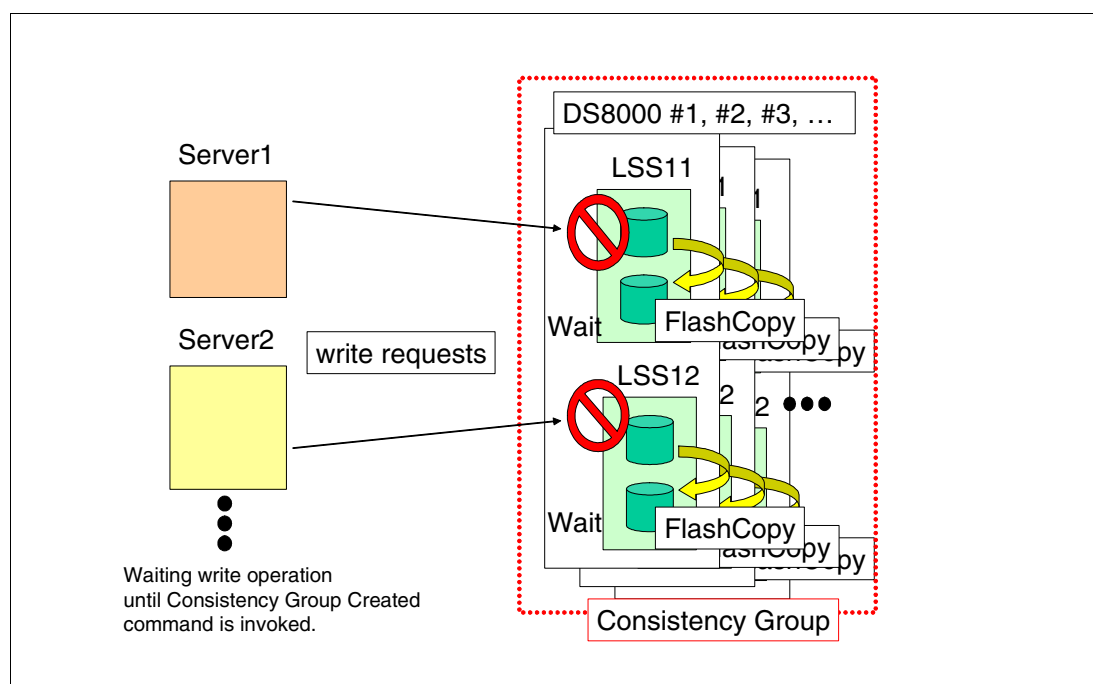


Figure 2-45 Consistency Group FlashCopy

**Important:** Consistency Group FlashCopy can create host-based consistent copies; they are not application-based consistent copies. The copies have *power-fail* or *crash* level consistency. This means that if you suddenly power off your server without stopping your applications and without destaging the data in the file cache, the data in the file cache can be lost and you might need recovery procedures to restart your applications. To start your system with Consistency Group FlashCopy target volumes, you might need the same operations as the crash recovery.

For example, if the Consistency Group source volumes are used with a journalized file system (such as AIX JFS) and the source LUNs are not unmounted before running FlashCopy, the **fsck** command will likely have to be run on the target volumes.



### ***Establish FlashCopy on existing Metro Mirror or Global Copy primary***

Use this option to establish a FlashCopy relationship where the target is also a Metro Mirror or Global Copy primary volume; see Figure 2-46. This option enables you to create full or incremental point-in-time copies at a local site and then use remote mirroring to copy the data to the remote site.

**Note:** You cannot perform FlashCopy from a source to a target if the target is also a Global Mirror primary volume.

For more information, see “Metro Mirror” on page 93 and in “Global Copy” on page 94.

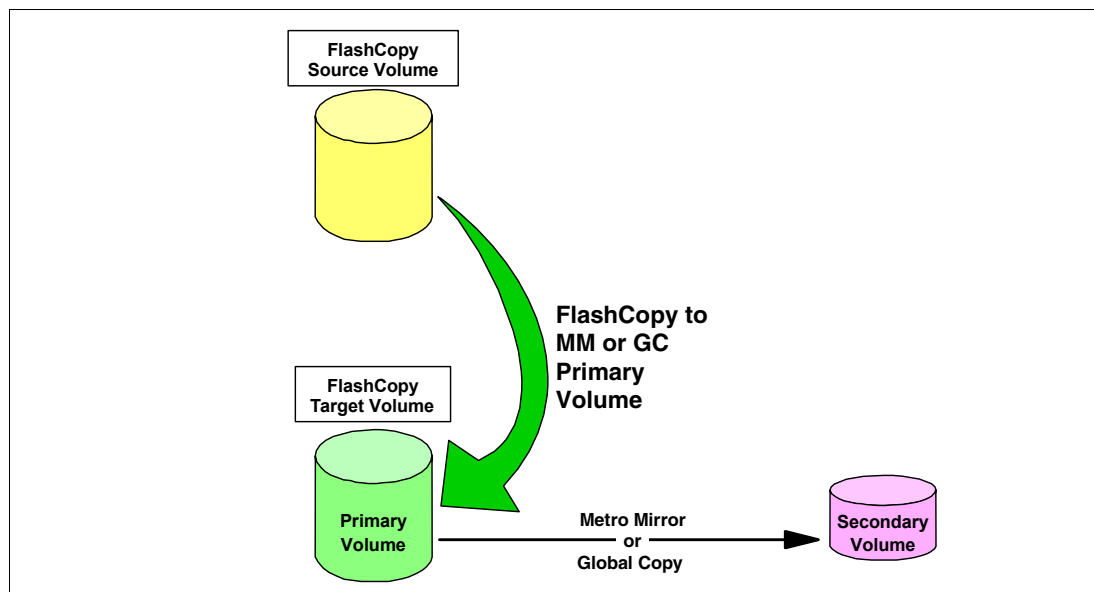


Figure 2-46 Establish FlashCopy on existing Metro Mirror (MM) or Global Copy (GC) primary

It took some time to initialize all the bitmaps that were needed for the scenario described. In DS8000 Licensed Machine Code (LMC) level 5.3.xx.xx.xx, the time to initialize the bitmaps has been greatly improved.

### ***Persistent FlashCopy***

Persistent FlashCopy allows the FlashCopy relationship to remain, even after the copy operation completes. You must explicitly delete the relationship.

### ***Inband commands over remote mirror link***

In a remote mirror environment, commands to manage FlashCopy at the remote site can be issued from the local or intermediate site and transmitted over the remote mirror Fibre Channel links. This process eliminates the need for a network connection to the remote site solely for the management of FlashCopy.

**Note:** This function is available only through the use of the DS CLI commands and not the DS Storage Manager GUI.

### ***Remote Pair FlashCopy***

A FlashCopy target can be the source of a Metro Mirror relationship, as illustrated in Figure 2-47 on page 92.

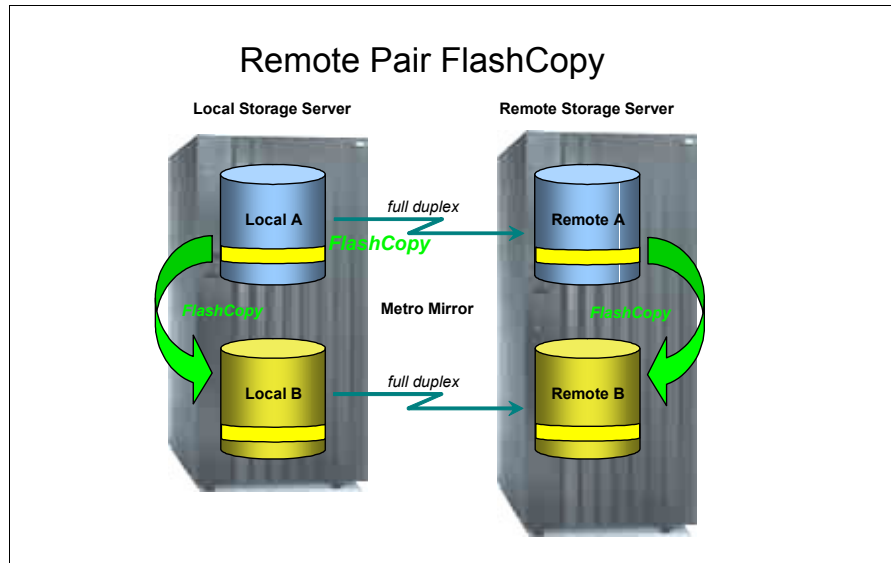


Figure 2-47 Remote Pair FlashCopy

To ensure the integrity of the mirror, the mirroring must be suspended while the tracks associated with the FlashCopy relationship are copied to the Metro Mirror secondary device. For many (mainframe) customers, any time that the mirror relationships in the customer environment are in a state other than fully synchronized (full duplex) is viewed as a loss of the mirror. If many FlashCopy operations are performed throughout the day (as is often the case with data set level copy operations), a situation might occur in which the Metro Mirror remote site is rarely, if ever, a true mirror of the local or production site.

Remote Pair FlashCopy (also called Preserve Mirror) is the solution to this problem in that it can also mirror the FlashCopy command issued at the local site to the remote site, therefore preserving the Metro Mirror duplex state, if the proper configuration exists.

For more information, see *IBM System Storage DS8000: Remote Pair FlashCopy (Preserve Mirror)*, REDP-4504.

**Note:** Remote Pair FlashCopy cannot be used with FlashCopy SE volumes as the target.

## IBM FlashCopy SE options

Many options available for standard FlashCopy (see “FlashCopy options” on page 87) are also available for FlashCopy SE. Only the options that differ are discussed in this section. We explain several options and capabilities in this section:

- ▶ Incremental FlashCopy
- ▶ Data set level FlashCopy
- ▶ Multiple relationship FlashCopy SE
- ▶ Consistency Group FlashCopy

### ***Incremental FlashCopy***

Because incremental FlashCopy implies a full-volume copy, and a full-volume copy is not possible in an IBM FlashCopy SE relationship, incremental FlashCopy is not possible with IBM FlashCopy SE.

### ***Data set level FlashCopy***

FlashCopy SE relationships are limited to full-volume relationships. As a result, data set level FlashCopy is not supported within FlashCopy SE<sup>9</sup>.

### ***Multiple relationship FlashCopy SE***

Standard FlashCopy supports up to 12 relationships and one of these relationships can be incremental. There is always overhead when doing a FlashCopy or any kind of copy within a storage subsystem. A FlashCopy onto a Space Efficient volume has more overhead because additional tables have to be maintained. All IBM FlashCopy SE relations are *nocopy* relations; incremental FlashCopy is not possible. Therefore, the practical number of IBM FlashCopy SE relationships from one source volume will be lower than 12. In your environment, be sure to test how many concurrent IBM FlashCopy SE relationships are acceptable from a performance standpoint.

### ***Consistency Group FlashCopy***

With IBM FlashCopy SE, consistency groups can be formed in the same way as with standard FlashCopy (see “Consistency Group FlashCopy” on page 89). Within a consistency group, there can be a mix of standard FlashCopy and IBM FlashCopy SE relationships.

## **Remote Mirror and Copy**

The Remote Mirror and Copy functions of the DS8000 are a set of flexible data mirroring solutions that allow replication between volumes on two or more disk storage systems. These functions are used to implement remote data backup and disaster recovery solutions.

The Remote Mirror and Copy functions are optional licensed functions of the DS8000:

- ▶ Metro Mirror
- ▶ Global Copy
- ▶ Global Mirror
- ▶ Metro Global Mirror

In addition, System z users can use the DS8000 for z/OS Global Mirror

In this section, we discuss these Remote Mirror and Copy functions.

For a more detailed and extensive discussion of these topics, see *DS8000 Copy Services for IBM System z*, SG24-6787.

Also, consider that some of the remote mirror solutions, such as Global Mirror, Metro/Global Mirror, or z/OS Metro/Global Mirror, integrate more than one licensed function. In this case, you have all of the required licensed functions.

### ***Metro Mirror***

Metro Mirror, previously known as Synchronous Peer-to-Peer Remote Copy (PPRC), provides real-time mirroring of logical volumes between two DS8000s that can be located up to 300 km from each other. It is a synchronous copy solution where write operations are completed on both copies (local and remote site) before they are considered complete.

---

<sup>9</sup> This implies that FlashCopy SE cannot be used by DB2 for utilities at the table space level.

Figure 2-48 illustrates the basic operational characteristics of Metro Mirror.

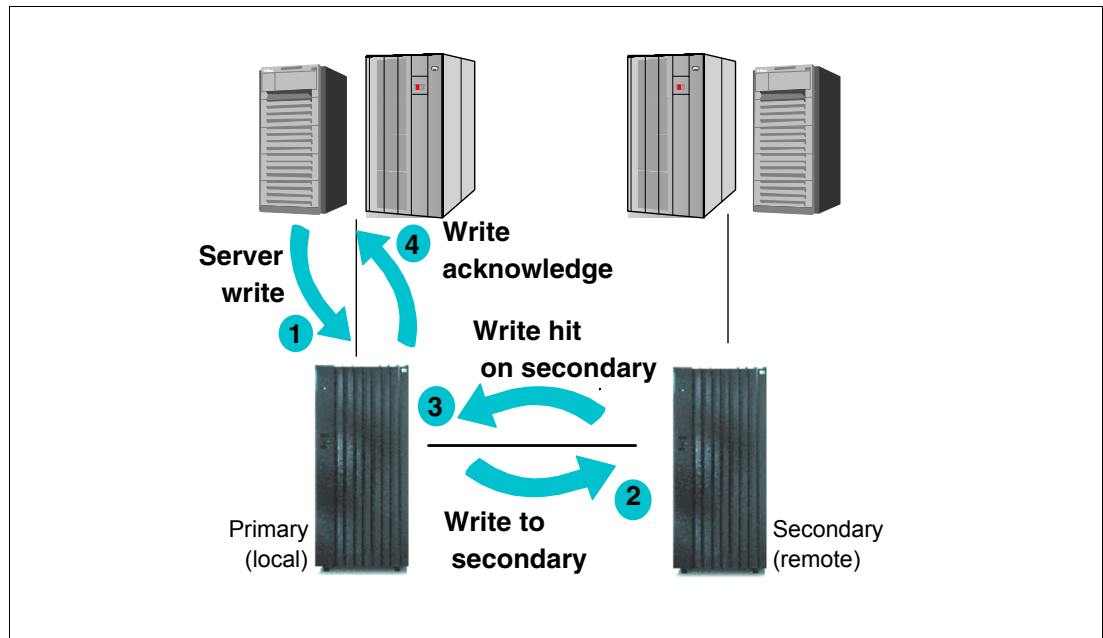


Figure 2-48 Metro Mirror basic operation

### Global Copy

Global Copy, previously known as Peer-to-Peer Remote Copy Extended Distance (PPRC-XD), copies data non-synchronously and over longer distances than is possible with Metro Mirror. When operating in Global Copy mode, the source volume sends a periodic, incremental copy of updated tracks to the target volume, instead of sending a constant stream of updates. This process causes less impact to application-writes for source volumes and less demand for bandwidth resources; it allows a more flexible use of the available bandwidth.

Global Copy does not keep the sequence of write operations. Therefore, the copy is a fuzzy copy, but you can make a consistent copy through synchronization (called a *go-to-sync operation*). After the synchronization, you can issue FlashCopy at the secondary site to make the backup copy with data consistency. After establishing FlashCopy, you can change the mode back to the non-synchronous mode. See Figure 2-49 on page 95.

An alternative method to acquire a consistent copy is to pause the applications until all changed data at the local site has drained to the remote site. When all consistent data is replicated to the remote site, suspend Global Copy, restart the applications, issue the FlashCopy, and then resume the non-synchronous (Global Copy) operation.

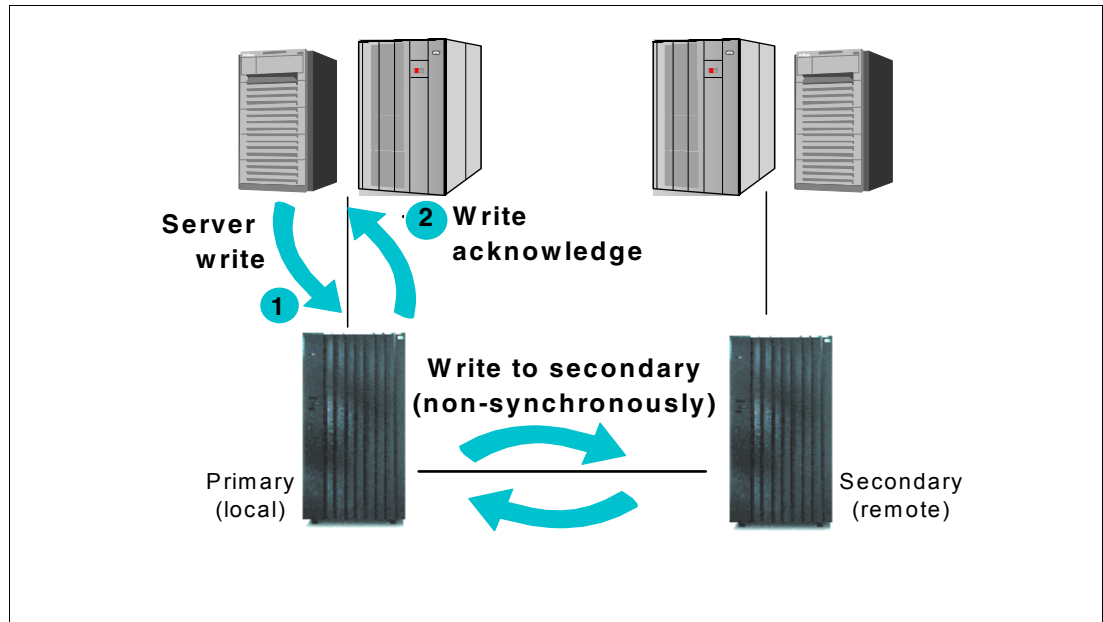


Figure 2-49 Global Copy basic operation

### Global Mirror

Global Mirror, previously known as Asynchronous PPRC, is a two-site, long distance, asynchronous, remote copy technology for both System z and open systems data. This solution integrates the Global Copy and FlashCopy technologies. With Global Mirror, the data that the host writes to the storage unit at the local site is asynchronously mirrored to the storage unit at the remote site. With special management steps, under control of the local master storage unit, a consistent copy of the data is automatically maintained on the storage unit at the remote site.

Global Mirror operations provide the following benefits:

- Support for virtually unlimited distances between the local and remote sites, with the distance typically limited only by the capabilities of the network and the channel extension technology

This *unlimited* distance enables you to choose your remote site location based on business needs and enables site separation to add protection from localized disasters.

- A consistent and restartable copy of the data at the remote site, created with minimal impact to applications at the local site
- Data currency where, for many environments, the remote site lags behind the local site typically 3 - 5 seconds, minimizing the amount of data exposure in the event of an unplanned outage

The actual lag in data currency that you experience will depend upon a number of factors, including specific workload characteristics and bandwidth between the local and remote sites.

- Dynamic selection of the desired recovery point objective (RPO), based upon business requirements and optimization of available bandwidth

- ▶ Session support whereby data consistency at the remote site is internally managed across up to eight storage units that are located across the local and remote sites.  
The number of managed storage units can be greater (SCORE/RPQ must be submitted).
- ▶ Efficient synchronization of the local and remote sites with support for failover and failback operations, helping to reduce the time that is required to switch back to the local site after a planned or unplanned outage

Figure 2-50 illustrates the basic operation characteristics of Global Mirror.

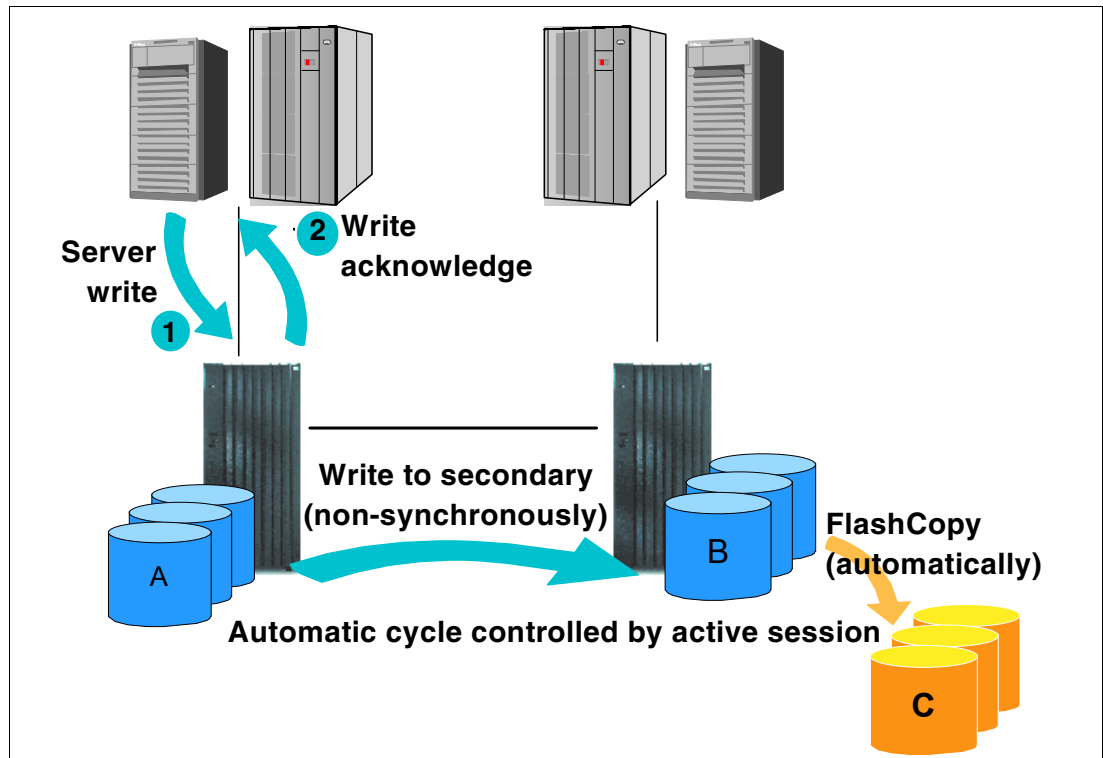


Figure 2-50 Global Mirror basic operation

Figure 2-51 on page 97 illustrates the basics of how Global Mirror works (everything in an automatic fashion under the control of the DS8000 microcode and the Global Mirror session).

You see that the A volumes at the local site are the production volumes and are used as Global Copy primary volumes. The data from the A volumes is replicated to the B volumes, which are the Global Copy secondary volumes. At a certain point in time, a Consistency Group is created using all of the A volumes, even if they are located in separate storage units. This approach has no application impact, because the creation of the Consistency Group is quick (on the order of milliseconds).

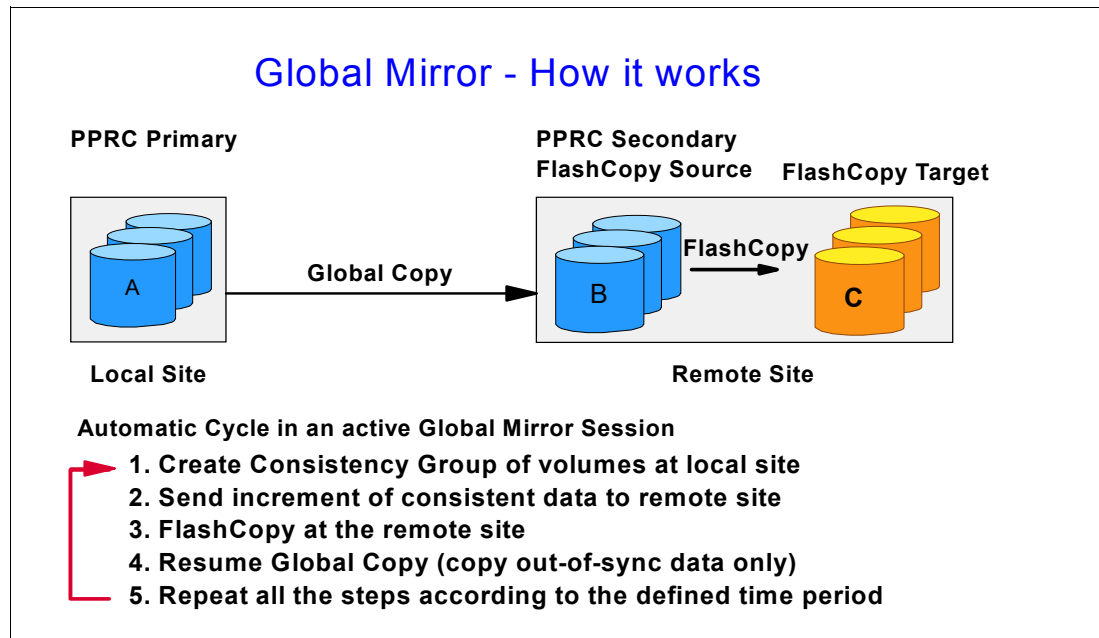


Figure 2-51 How Global Mirror works

After the Consistency Group is created, the application-writes can continue updating the A volumes. The increment of the consistent data is sent to the B volumes using the existing Global Copy relationships. When the data reaches the B volumes, it is copied (FlashCopy) to the C volumes.

After this step is complete, a consistent set of volumes have been created at the secondary site. For this brief moment only, the B volumes and the C volumes are equal in their content. Because the B volumes, except at the moment of doing the FlashCopy, usually contain a *fuzzy* copy of the data, the C volumes are used to hold the last consistent point-in-time copy of the data while the B volumes are being updated by Global Copy. Therefore, you need the B and the C volume to build consistent data.

Although the data at the remote site is current within typically 3 - 5 seconds, this recovery point depends on the workload and bandwidth that is available to the remote site.

With its efficient and autonomic implementation, Global Mirror is a solution for disaster recovery implementations where a consistent copy of the data must be kept at all times at a remote location that can be separated by a very long distance from the production site. For more information about the functionality of Copy Services, see *DS8000 Copy Services for IBM System z*, SG24-6787.

### **Metro Global Mirror**

Metro Global Mirror is a three-site, multi-purpose, replication solution for both System z and open systems data. Local site (site A) to intermediate site (site B) provides high availability replication using Metro Mirror, and intermediate site (site B) to remote site (site C) supports long distance disaster recovery replication with Global Mirror. See Figure 2-52 on page 98.

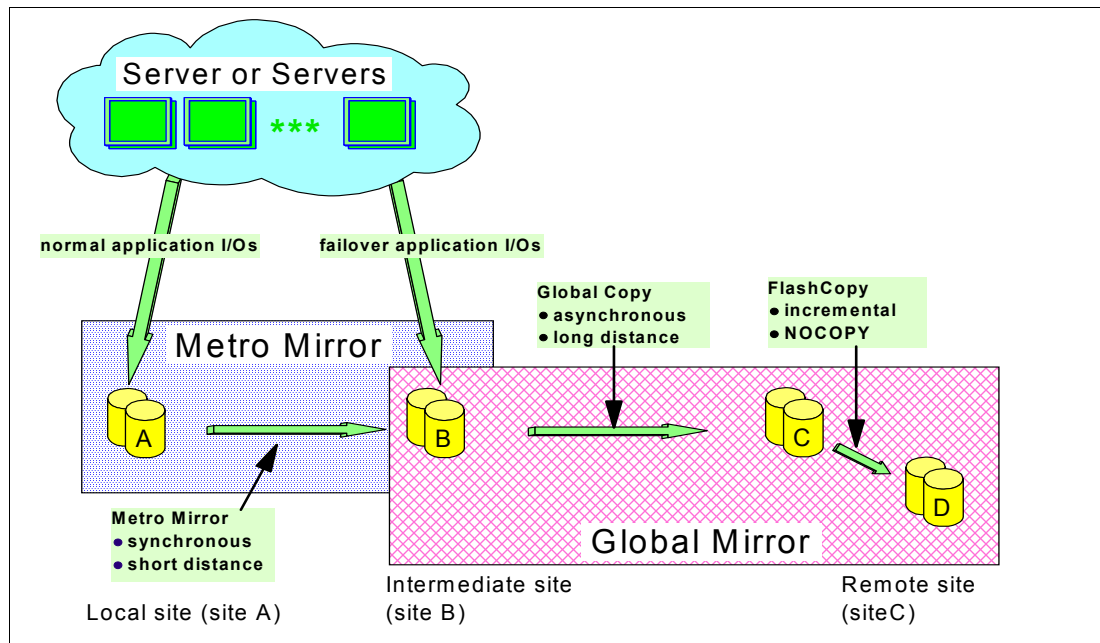


Figure 2-52 Metro/Global Mirror elements

Both Metro Mirror and Global Mirror are well established replication solutions. Metro/Global Mirror combines Metro Mirror and Global Mirror to incorporate the best features of the two solutions:

► Metro Mirror:

- Synchronous operation supports zero data loss.
- The opportunity to locate the intermediate site disk subsystems close to the local site allows use of intermediate site disk subsystems in a high availability configuration.

**Note:** Metro Mirror can be used for distances of up to 300 km, but when used in a Metro/Global Mirror implementation, a shorter distance might be more appropriate in support of the high availability functionality.

► Global Mirror:

- Asynchronous operation supports long distance replication for disaster recovery.
- Global Mirror methodology allows for no impact to applications at the local site.
- This solution provides a recoverable, restartable, consistent image at the remote site with an RPO typically in the 3 - 5 second range.

Figure 2-53 on page 99 gives an overview of the Metro/Global Mirror process. If you understand the processes of the components, understanding the Metro/Global Mirror process is easier to understand. One important consideration is that the intermediate site volumes (site B volumes) are special, because they act as both Global Mirror (GM) *source* and Metro Mirror (MM) *target* volumes at the same time.



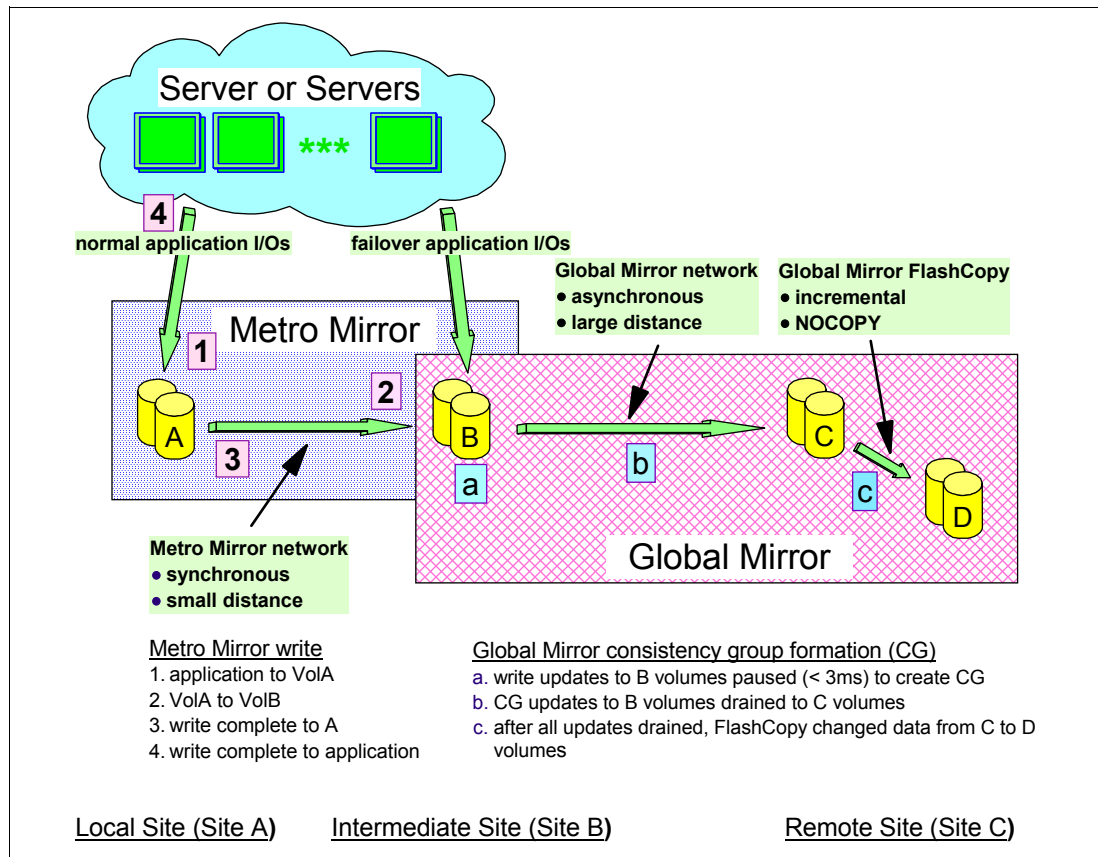


Figure 2-53 Metro/Global Mirror overview diagram

The local site (site A) to intermediate site (site B) component is identical to Metro Mirror. Application write operations are synchronously copied to the intermediate site before write complete is signaled to the application. All write operations to the local site volumes in the mirror are treated in exactly the same way.

The intermediate site (site B) to remote site (site C) component is identical to Global Mirror, with the following exceptions:

- The writes to intermediate site volumes are Metro Mirror secondary writes and not application primary writes.
- The intermediate site volumes are both GM *source* and MM *target* at the same time.

The intermediate site disk subsystems are collectively paused by the Global Mirror Master disk subsystem to create the Consistency Group (CG) set of updates. This *pause* normally takes 3 ms every 3 - 5 seconds. After the CG set is formed, the Metro Mirror write operations from local site (site A) volumes to intermediate site (site B) volumes are allowed to continue. Also, the CG updates continue to drain to the remote site (site C) volumes. The intermediate site to remote site drain should take only a few seconds to complete.

After all updates are drained to the remote site, all changes since the last FlashCopy from the C volumes to the D volumes are logically (NOCOPY) copied (FlashCopy) to the D volumes. After the logical FlashCopy is complete, the intermediate site to remote site Global Copy data transfer is resumed until the next formation of a Global Mirror CG. The process described is repeated every 3 - 5 seconds if the interval for Consistency Group formation is set to zero. Otherwise, it is repeated at the specified interval plus 3 - 5 seconds.

The Global Mirror processes are discussed in greater detail in *DS8000 Copy Services for IBM System z*, SG24-6787.

## z/OS Global Mirror

z/OS Global Mirror, previously known as Extended Remote Copy (XRC), is a copy function available for the z/OS operating system. It involves a System Data Mover (SDM) that is found only in z/OS. The z/OS Global Mirror maintains a consistent copy of the data asynchronously at a remote location, and can be implemented over unlimited distances. It is a combined hardware and software solution that offers data integrity and data availability and can be used as part of business continuance solutions, for workload movement, and for data migration. z/OS Global Mirror function is an optional licensed function of the DS8000.

Figure 2-54 illustrates the basic operational characteristics of z/OS Global Mirror.

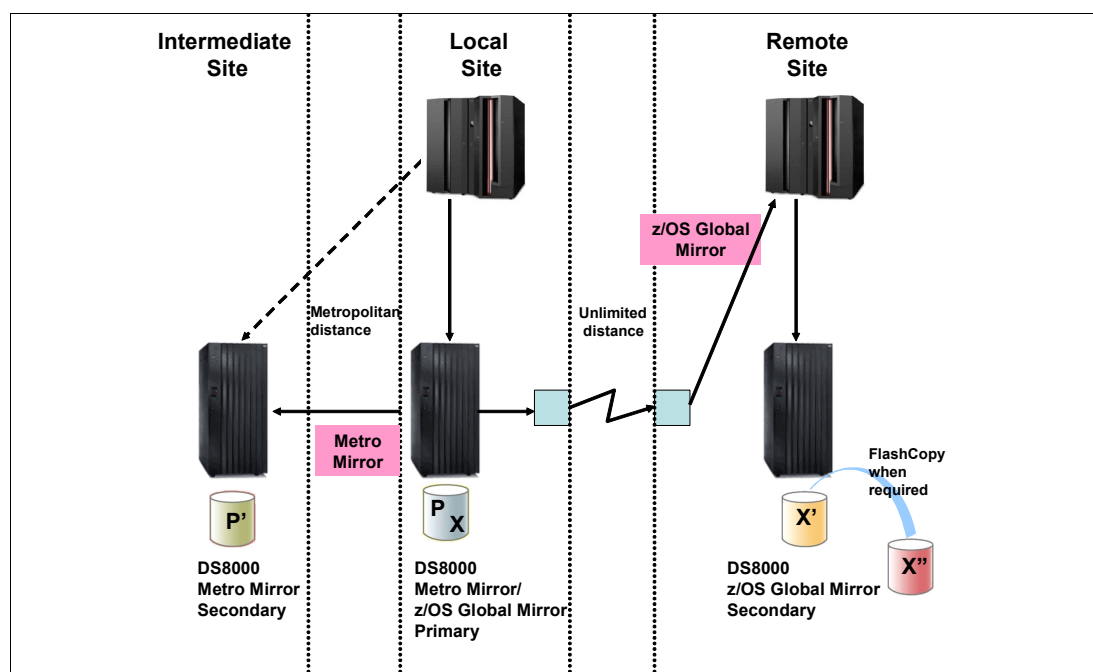


Figure 2-54 z/OS Metro Global Mirror

In this section, we summarize the use of and considerations for the set of Remote Mirror and Copy functions available with the DS8000 series:

- ▶ Metro Mirror
- ▶ Global Copy
- ▶ Global Mirror
- ▶ z/OS Global Mirror on zIIP
- ▶ Compression

### Metro Mirror

Metro Mirror is a function for synchronous data copy at a distance. The following considerations apply:

- ▶ No data loss exists and Metro Mirror allows for rapid recovery for distances up to 300 km.
- ▶ A slight performance impact exists for write operations.

### ***Global Copy***

Global Copy is a function for non-synchronous data copy at very long distances, which is limited only by the network implementation. The following considerations apply:

- ▶ It can copy your data at nearly an unlimited distance, making it suitable for data migration and daily backup to a remote distant site.
- ▶ The copy is normally *fuzzy* but can be made consistent through a synchronization procedure.

To create a consistent copy for Global Copy, you need a go-to-sync operation; that is, synchronize the secondary volumes to the primary volumes. During the go-to-sync operation, the mode of remote copy changes from a non-synchronous copy to a synchronous copy. Therefore, the go-to-sync operation might affect performance of your application system. If the data is heavily updated and the network bandwidth for remote copy is limited, the time for the go-to-sync operation becomes longer. An alternative method to acquire a consistent copy is to pause the applications until all changed data at the local site has drained to the remote site. When all consistent data is at the remote site, suspend Global Copy, restart the applications, issue the FlashCopy, and then return to the non-synchronous (Global Copy) operation.

### ***Global Mirror***

Global Mirror is an asynchronous copy technique; you can create a consistent copy in the secondary site with an adaptable recovery point objective (RPO). RPO specifies how much data you can afford to re-create if the system must be recovered. The following considerations apply:

- ▶ Global Mirror can copy to nearly an unlimited distance.
- ▶ Global Mirror is scalable across the storage units.
- ▶ Global Mirror can realize a low RPO if enough link bandwidth exists; when the link bandwidth capability is exceeded with a heavy workload, the RPO might grow.
- ▶ Global Mirror causes only a slight performance and RPO (data loss) impact to your application system.

### ***z/OS Global Mirror on zIIP***

The IBM z9 Integrated Information Processor (zIIP) is a special engine available on z10, z9 EC, and z9 BC servers. z/OS provides the ability to utilize these processors in order to handle eligible workloads from the System Data Mover (SDM) in an z/OS Global Mirror (zGM) environment.

The normal SDM workload can be divided into service request block (SRB) service time and task control block (TCB) service time. The SRB mode service time is not eligible for a zIIP engine. Most of the SDM workload is in TCB mode, which has been moved to the zIIP processor. It is not necessary to run the SDM LPARs on processors that are software license charged, since zIIP capacity is not imposed for software charges.

The zIIP assisted z/OS Global Mirror feature is available with the following items:

- ▶ IBM System z9 or z10 server with one or more zIIP processors
- ▶ IBM System storage DS8000, or any storage controller supporting z/OS GM
- ▶ z/OS V1.8 and later, with APAR OA23174

This feature is enabled through the zGM PARMLIB member setting, `zIIPEnable` (YES).

**Note:** More zIIPs cannot be active than CPs.

### ***Compression***

The IBM disk storage subsystem does not have an internal data compression on disk like the RVA. However, the tape storage subsystem provides data compression on tape volumes.

## **2.13 Solid-state drives with DS8000**

Solid-state drives (SSDs) are high I/O per second (IOPS) class-enterprise storage devices that are targeted at business-critical production applications, which can benefit from high level of fast-access storage.

In addition to better IOPS performance, SSDs offer a number of potential benefits over electromechanical hard disk drives (HDDs), including better reliability, lower power consumption, less heat generation, and lower acoustical noise.

This section gives a brief overview of SSDs, available for all DS8000 models as an optional, RPQ-based feature and the IBM Easy Tier:

- ▶ Solid-state drives overview
- ▶ Easy Tier

### **2.13.1 Solid-state drives overview**

Besides an ever increasing demand for more storage capacity, many businesses constantly require faster storage performance for their high-end, business-critical applications.

The HDD technology has improved dramatically over the years, sustaining more IOPS and increased throughput, and providing higher storage density at a lower price. However, because of its moving and spinning parts, there are inherent physical limitations to the response times (access time and latency) that an HDD can achieve.

These limitations and the continuing demand for better response times and more throughput have created the need for a different storage technology that avoids seeks and rotational delays. This technology, known as solid-state drives, can provide access times measured in microseconds.

SSDs use semiconductor devices (solid-state memory) to store data and have no moving parts. SSDs are in fact not new and have existed for a while. They were initially based on dynamic random access memory (DRAM) and often referred to as RAM-disks. Although they had good performance, they were also extremely expensive and most businesses could not justify the cost.

The technology has evolved and SSDs are now based on non-volatile flash memory, as is used in USB flash drives and camera memory cards. Flash memory is much cheaper than DRAM although still more costly per gigabyte than high-end HDDs. The technology is still improving and the cost of SSDs continues to drop, allowing them to compete more strongly with electromechanical disks.

### **SSD implementation in DS8000**

Starting with Licensed Machine Code 5.40.20.xx, the IBM DS8000 series provides support for SSDs. Initially drives are available in 73 GB and 146 GB capacities and are Single Level Cell-Flash (SLC-Flash) drives with dynamic and static wear-leveling, bad-block mapping, and error detection code/error correction code (EDC/ECC).

These high-IOPS enterprise-class storage devices are classified as Tier 0 storage according to the following tiers of enterprise storage systems:

- ▶ Tier 0: Enterprise SSD (Fibre Channel SSD, SAS SSD, which is serial attached SCSI SSD)
- ▶ Tier 1: Enterprise HDD (Fibre Channel HDD, SAS HDD)
- ▶ Tier 2: SATA SSD and SATA HDD (SATA, which is Serial ATA)
- ▶ Tier 3: Tape

The SSDs are optional to the DS8000 series and are available with feature numbers 6xxx.

SSDs have the same form factor as the regular spinning drives available for the DS8000 and they come pre-installed in a regular DS8000 disk enclosure, half populated with 8 drives.

### **Initial configuration recommendations**

Use the following guidelines for initially configuring a DS8000 with SSDs:

- ▶ For new (manufacturing order) DS8000 systems to have SSDs installed, the system must be ordered with RPQ 8S1027. SSDs are available, also through RPQ on field systems, to ensure proper placement for best performance.
- ▶ Configure 16 SSDs per device-adaptor (DA) pair, with a maximum of 32. The reason is to maintain adequate performance without saturating the DA. The SSDs that are supported in the DS8000 are so fast that the DA can become the performance bottleneck on some random workloads. Therefore, use only 16 SSDs per DA pair to get the maximum performance from each SSD. (The DS8000 supports up to eight DA pairs.)
- ▶ Considering that the DS8000 can have a maximum of eight DA pairs installed, the current maximum number of SSDs in a DS8000 should be limited to 128.
- ▶ SSDs can be intermixed with traditional drives, either on the same DA pair, or on separate DA pairs within the DS8000. Do not intermix SSDs and HDDs on the same DA pair, because this negatively affects the performance benefit of having SSDs
- ▶ When intermixed with spinning drives (HDDs), the total number of drives (SSDs and HDDs combined) is limited to 256 in model 931, or 512 in models 932 and 9B2, which limits the expansion frame configurations to three frames (one base frame and two expansion frames).
- ▶ Use RAID 5 for SSD drives. RAID 6 and RAID 10 are not recommended and initially not supported on the DS8000 for SSD drives. RAID 6 consumes too much expensive flash space for little benefit (and lower performance), and RAID 10 provides little performance improvement because SSDs are already so responsive to random reads.
- ▶ SSDs are not available as Standby Capacity on Demand disk drives.

## **2.13.2 Easy Tier**

IBM DS8700 disk storage system now includes a technology invented by IBM Research that can make managing data in tiers easier and more economical. The IBM System Storage Easy Tier feature uses ongoing performance monitoring to move only the most active data to faster SSDs, which can eliminate the need for manual storage tier policies and help reduce costs. By automatically placing clients' most critical data on SSDs, Easy Tier provides quick access to data so it can be analyzed for insight as needed, to provide competitive advantage.

This section describes the key features of Easy Tier on the DS8000.

### Sub-LUN automatic movement

Today, *extent pools* can be either SSD-only or HDD-only. Easy Tier introduces *mixed* SSD and HDD extent pools. The hottest extents are moved to SSD, and cooler extents are moved down to HDD. The support applies to both fixed-block architecture (FBA) LUNs and count key data (CKD) volumes, which means that an individual LUN or CKD volume can have some of its 1GB extents on SSD and other extents on FC or SATA disk.

### Entire-LUN Manual Relocation

Entire-LUN Manual Relocation (ELMR) can relocate an entire LUN nondisruptively from any extent pool to any other extent pool. You can relocate LUNs from an SSD-only or HDD-only pool over to a new Easy Tier-managed *mixed* pool, or take a LUN out of Easy Tier management by moving it to an SSD-only or HDD-only pool. Of course, this support also applies to both FBA LUNs and CKD volumes.

This feature can also be used to relocate LUNs and CKD volumes from FC to SATA pools, from RAID 10 to RAID 5 pools, and so on.

### Pool mergers

If you already have SSD-only and HDD-only pools and want to use Easy Tier, you can now merge pools to create a mixed pool.

### SSD mini-packs

Previously, you had to buy 16 solid-state drives at a time, called mega-packs. Now, you can choose to buy only eight SSDs at a time; these are called mini-packs. Moving as little as 10% of your data from Fibre Channel disk to SSD with Easy Tier can result in up to a 300 - 400% performance improvement. Results of the formal SPC-1 benchmark are available from the following location:

[http://www.storageperformance.org/benchmark\\_results\\_files/SPC-1/IBM/A00092\\_IBM\\_DS8700\\_EasyTier-SSDs/a00092\\_IBM\\_DS8700\\_EasyTier-SSDs\\_SPC1\\_executive-summary.pdf](http://www.storageperformance.org/benchmark_results_files/SPC-1/IBM/A00092_IBM_DS8700_EasyTier-SSDs/a00092_IBM_DS8700_EasyTier-SSDs_SPC1_executive-summary.pdf)

### Storage Tier Advisor Tool (STAT)

If you do not have SSD yet or you are not sure how beneficial Easy Tier is for your data center, use the IBM Storage Tier Advisor Tool. The tool can analyze your extents and estimate how much benefit you will derive if you implement Easy Tier with various amounts of SSD.

## 2.14 Tapes: Storage Tier 3 in z/OS environment

As described in “IBM enterprise tape systems” on page 7, the last tier in the storage hierarchy is tape. IBM has several tape solutions in the portfolio: non-virtualized and virtualized. They are summarized in Figure 2-55 on page 105.

IBM enterprise class tape products are designed to offer high performance, availability, reliability, and capacity to help meet the needs of customers seeking enterprise class storage solutions for mass storage, data archiving, enterprise backup, and disaster recovery. IBM enterprise class storage products provide on demand solutions for business.

#### ► TS7700 Virtualization Engine

The IBM Virtualization Engine TS7700 (TS7700 Virtualization Engine) is a family of mainframe virtual-tape solutions that are designed to optimize tape processing. With one solution, the implementation of a fully integrated tiered storage hierarchy of disk and tape utilizes the benefits of both technologies to help enhance performance and provide the

capacity needed for today's tape processing requirements. Deploying this innovative subsystem can help reduce batch processing time, total cost of ownership and management overhead.

▶ TS7680 ProtecTIER Deduplication Gateway

The IBM System Storage® TS7680 ProtecTIER® Deduplication Gateway for System z® combines a virtual tape library solution, with IBM's unique and patented HyperFactor® deduplication technology and integrated native replication technology to provide users an optimal disk-based target for Systems z applications that traditionally use tape.

▶ TS3500 Tape Library

The IBM TS3500 Tape Library supports System z® through the IBM 3953 Tape System (3953 tape system) or the IBM Virtualization Engine TS7740, which enable System z hosts to access the TS3500 Tape Library cartridge inventory and allow connection to TS1130, TS1120 and IBM TotalStorage 3592 Model J1A tape drives.

▶ TS1130 Tape Drive

The IBM TS1130 Tape Drive offers high-performance flexible data storage with support for data encryption. The TS1130 Tape Drive can be installed in the TS3500 Tape Libraries, IBM TotalStorage 3494 Tape Libraries, IBM racks that enable stand-alone installation, and IBM 3952 Tape Frames Model C20 (3952C20 frame) attached to a Sun 9310 library. When installed in a TS3500 Tape Library, it also attaches to the TS7740 Virtualization Engine.

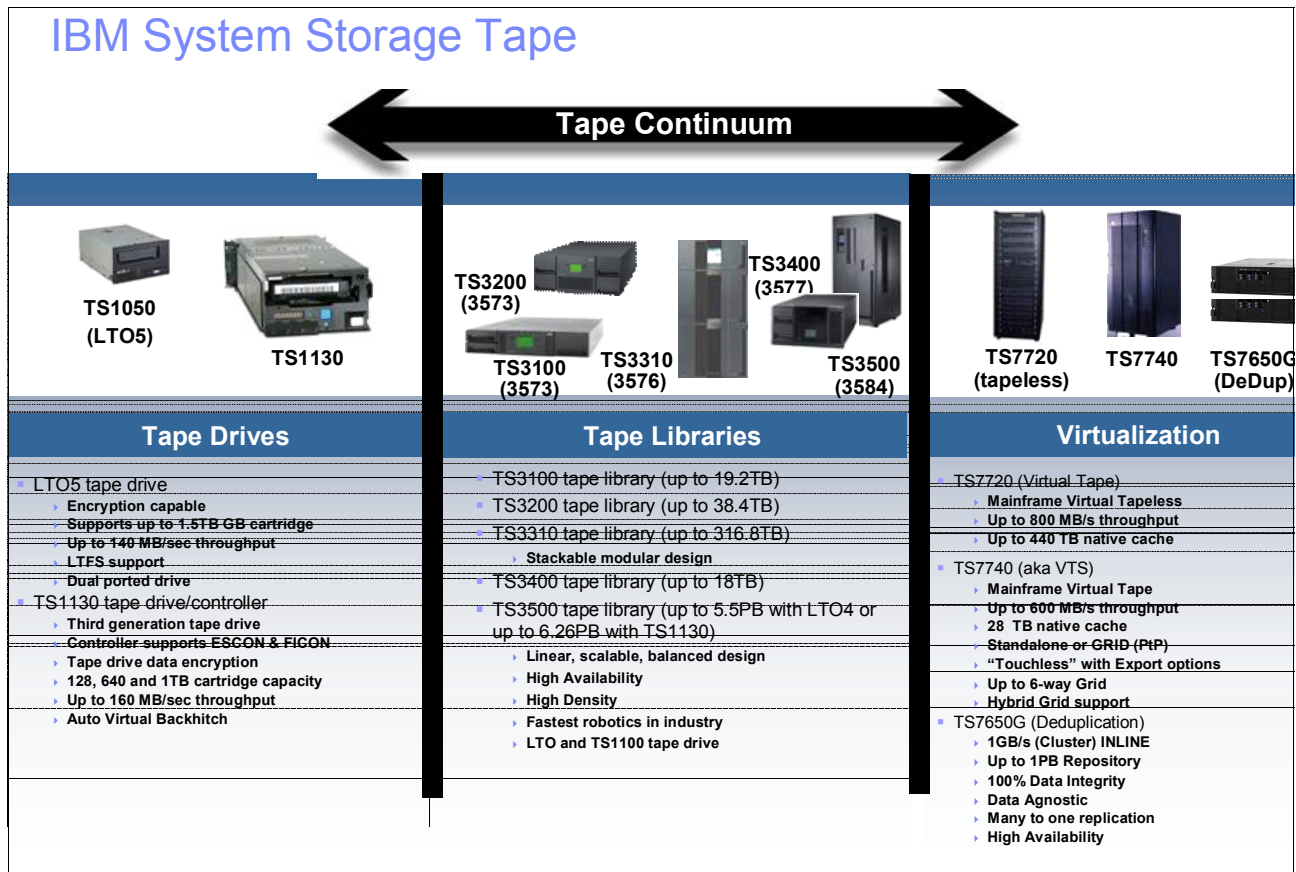


Figure 2-55 IBM Tape solutions

Over the last decade, tape solutions received a virtualization technology such as the DASD solutions. Today, we can still have an automated tape library (ATL) attached to System z without any virtualization functionality but also virtualized. The same effect is in the DASD domain. The virtualization is transparent to the application but it allows getting access to larger tape storage and is also faster, as before. This section briefly describes the newer tape solutions.

DB2 can benefit from such virtualized tape solutions also.

### 2.14.1 Tape virtualization

Tape virtualization was introduced as a response to the under utilization of tape capacity. Because many applications are not able to fill the high capacity media of today's tape technology, you can accumulate a large number of under utilized cartridges, wasting a lot of space on your physical media and requiring an excessive number of cartridge slots in your tape automation system. *Tape virtualization* fully utilizes the capacity of current tape technology by using a disk buffer as intermediate storage before writing client data to high capacity tape cartridges. However, System z virtual tape subsystems emulate the function and operation of IBM 3490 Enhanced Capacity (3490E) tape drives. These emulated tape drives are called *virtual tape drives*, and to an attached host, they appear identical to physical tape drives. Emulation is transparent to the host and to applications. Another benefit of tape virtualization is the large number of drives available to applications.

Each subsystem TS7700 provides 256 virtual tape devices. If you have a situation where applications contend for tape drives and jobs have to wait because no physical tape drive is available, tape virtualization can efficiently address these issues by providing a large number of virtual tape drives.

Figure 2-56 illustrates the main components that are used to create a complete virtualization solution.



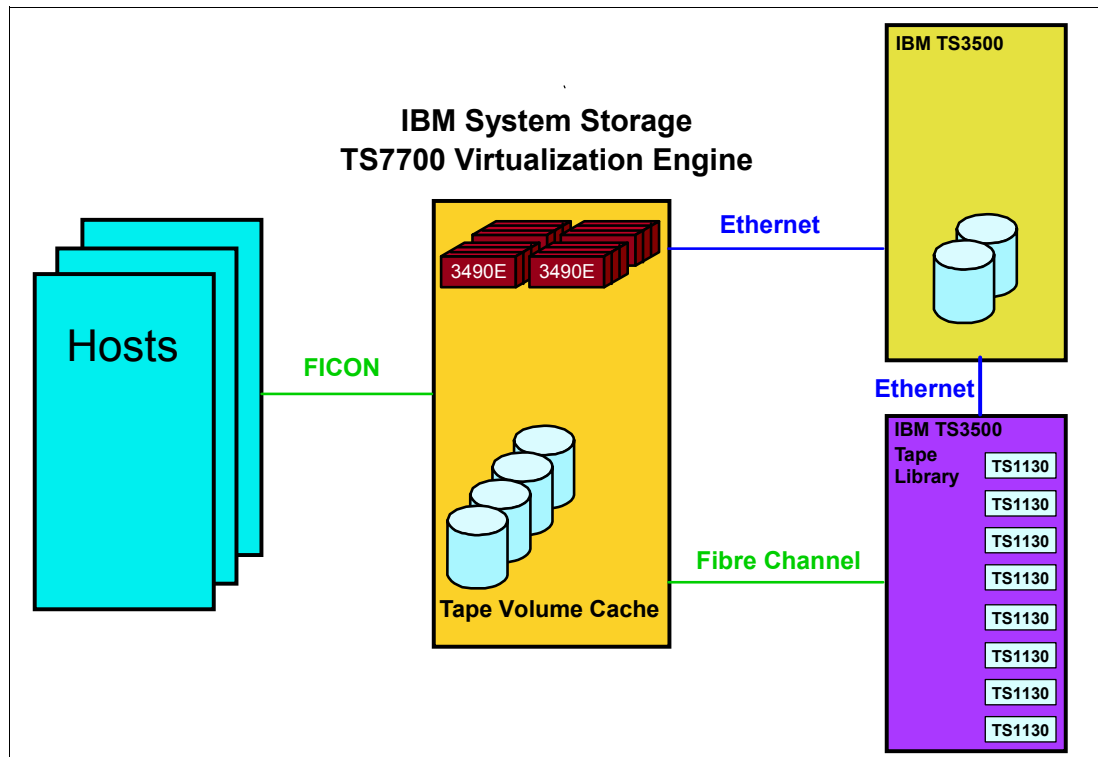


Figure 2-56 Main components of a tape virtualization solution

Data written to or read through a virtual tape drive resides in a virtual volume on a portion of the virtualization subsystem's disk storage media called the *Tape Volume Cache (TVC)*. A *virtual volume* shows the same characteristics as a Cartridge System Tape or an Enhanced Capacity Cartridge System Tape.

The TVC's associated storage management software manages the virtual volumes and the movement of data between the TVC and the IBM System Storage TS1120 and TS1130 tape drives associated with the virtualization subsystem.

A migration occurs when a virtual volume is moved from the TVC to a physical tape cartridge to free space in the TVC for new data. The virtual volume is now a logical volume. A recall occurs when a logical volume is moved from a physical tape cartridge back to the TVC. The logical volume again becomes a virtual volume. One or more logical volumes can reside on a physical tape cartridge (also called a *physical volume*). A tape cartridge containing more than one logical volume is called a *stacked volume*. Each virtualization subsystem is assigned a range of VOLSERs for use with its physical volumes. The virtualization subsystem maintains the relationship between logical volumes and the physical volumes on which they reside.

The TS7700 Controller maintains a database that is based on the VOLSERs of the physical volumes and also stores the logical volumes associated with the physical volumes.

The IBM Tape Virtualization solution for System z servers, the IBM Virtualization Engine TS7700, provides high availability solutions. Up to four Virtualization Engines, TS7720 or TS4440 or a combination of both, can be connected for data redundancy and high availability. More specifically, up to three TS7700 Virtualization Engines can be connected into a Multi Cluster Grid.

The following list summarizes the key benefits that you can expect from tape virtualization:

- High availability and disaster recovery configurations

- ▶ Fast access to data through caching on disk
- ▶ Utilization of current tape drive, tape media, and tape automation technology
- ▶ Transparent migration to new tape technology
- ▶ Capability of filling high capacity media 100%
- ▶ Large number of tape drives available for concurrent use
- ▶ No additional software required
- ▶ Reduced total cost of ownership (TCO)

### ***Tape virtualization design***

In the following sections, we provide short descriptions of the concepts that we use throughout this section. You have to consider the differences between 3494 VTS and the TS7700 Virtualization Engine.

### ***Tape Volume Cache (TVC)***

TVC is disk space where logical volumes are stored. The TVC is under complete and exclusive control of the virtualization subsystem. TVC capacity can be adjusted so that it can hold more virtual volumes than only those currently associated with virtual tape drives. The TVC storage management software adjusts capacity as necessary. After an application modifies and closes a virtual volume, it is copied by the storage management software to a physical volume as a background process. The virtual volume remains active until its disk space is needed to satisfy another mount request. Leaving the virtual volume in place in the TVC allows for fast access to its data when subsequently requested, much faster than if a physical volume needed to be mounted in a tape drive and read. When the host requests to mount the volume, the physical volume is recalled to the TVC as a virtual volume if the original virtual volume is no longer resident in the TVC.

### ***Physical drives***

The physical tape drives that are used by the TS7700 Virtualization Engine, IBM System Storage TS1120 and TS1130 Tape, are installed in the IBM TS3500 Tape Library. The physical tape drives are not addressable by any attached host system. They are completely under the control of the tape virtualization subsystem.

### ***Stacked volume***

The physical cartridges used by the tape virtualization subsystem to store logical volumes are completely under its control and are not known to the hosts. The physical volumes are called *stacked volumes*. The stacked volumes must have unique machine-readable VOLSERs and external labels just as any other cartridges in a tape library.

Through the TS7700 Management Interface and the TS3500 Tape Library Specialist, you define which physical cartridges are to be used by the tape virtualization subsystem. Logical volumes stored on those cartridges are mapped by the internal storage management software. When you use pooling, your stacked volumes can be assigned to individual pools. Logical volumes can then be assigned to the stacked volume pools.

### ***Virtual drives***

From a host perspective, a tape virtualization subsystem looks like multiple logical IBM 3490E tape control units, each with 16 drives attached through FICON channels. Virtual tape drives are defined like physical IBM 3490 controller addresses through the hardware configuration definition (HCD). Up to 256 virtual drives are supported.

Using regular operating system (OS) commands, you can manage virtual drives just as you manage real tape drives.

### ***Virtual volumes***

A *virtual volume* is created in the TVC when the host writes data to the tape virtualization subsystem. All host interaction with tape data in a tape virtualization system is through virtual volumes and virtual tape drives.

Each virtual volume has the same characteristics as a real volume, except for capacity. With a tape virtualization subsystem, a large volume size option is available, which increases the maximum native volume size up to 4000 MB. The end-of-volume is signaled when the total number of bytes written into the TVC after compression has reached 400 MB for an emulated cartridge system tape (CST), 800 MB for an emulated enhanced capacity cartridge system tape (ECCST) volume, or 1000, 2000, or 4000 MB using the larger volume size options.

With data compression based on the Ziv-Lempel algorithm (IBMLZ1) by the FICON channel card in a tape virtualization subsystem, the maximum actual host data stored on a virtual CST or ECCST volume can vary from 1.2 GB up to 12 GB (assuming a 3:1 compression ratio). The default logical volume sizes of 400 MB or 800 MB, still used at insert-time, can be overwritten at every individual scratch mount by the use of a Data Class construct.

### ***Logical volumes***

When a virtual volume is copied from the TVC to a physical tape cartridge, it becomes a *logical volume*. When a logical volume is moved from a physical cartridge to the TVC, the process is called *recall*, and the volume becomes a virtual volume again.

Although tape virtualization subsystems emulate a 3490E tape of a specific size, 400, 800, 1000, 2000, or 4000 MB, the space used in the TVC is no more than that needed for the number of bytes of data written to the virtual volume. When the virtual volume is written to the physical tape, it uses only the space occupied by the data. This means that neither the TVC nor the physical tapes are partitioned in any way into 400 MB, 800 MB, 1000 MB, 2000 MB, or 4000 MB segments. As virtual volumes are copied from the TVC to a physical cartridge, they are stacked on the cartridge end to end, taking up only the space written by the host application. This arrangement maximizes the utilization of a cartridge's storage capacity.

You can define up to 1 000 000 virtual volumes per stand alone TS7700 Virtualization Engine. A Multi Cluster Grid can have the same number of virtual volumes as a stand-alone cluster (1 000 000).

### ***Data compression***

IBM virtualization subsystems include the data compression capability. Data is compressed when written into the TVC. Whether tape drive compression is switched on is determined by whether it provides any additional benefit over the compression in the TVC.

## **2.14.2 TS7700 Virtualization Engine**

The IBM System Storage Virtualization Engine TS7700 represents the fourth generation of IBM Tape Virtualization for mainframe systems and replaced the highly successful IBM TotalStorage Virtual Tape Server (VTS) in 2006.

The TS7700 Virtualization Engine is designed to provide improved performance and capacity to help lower the total cost of ownership for tape processing. It introduces a new modular, scalable, high-performing architecture for mainframe tape virtualization. It integrates the advanced performance, capacity, and data integrity design of the IBM TS1130 tape drives with high-performance disk and a new advanced IBM System p® server to form a storage hierarchy managed by robust storage management firmware with extensive self-management capabilities.

TS7700 Virtualization Engine is built on a distributed node architecture.

The nodes perform either virtualization (vNode) or hierarchical data storage management (hNode). Based on the node architecture, a vNode or an hNode can run on separate virtualization hardware or can be combined to run on the same virtualization hardware. When a vNode and an hNode are combined and run on the virtualization hardware, this combination of nodes is referred to as a gNode (general node).

### ***Grid configuration***

Up to four TS7700 clusters can be interconnected to provide a disaster recovery/high availability solution that is called a *Multi Cluster Grid*. Each TS7700 cluster has 256 virtual 3490E drives. The Grid enablement feature must be installed on the TS7700 Virtualization Engines, and an Ethernet connection must be established between the clusters. Each cluster has two 1 Gb Ethernet adapters and uses TCP/IP for communication to the other TS7700 clusters.

Logical volume attributes and data are replicated across the clusters in a grid configuration to ensure the continuation of production work if a single cluster becomes unavailable. Any data replicated between the clusters is accessible through any of the other clusters in a grid configuration. More specifically, all vNodes in a grid have direct access to all logical volumes in the grid, either from the local cluster's TVC through Fibre Channel or from a remote cluster's TVC through a wide area network (WAN) connection. During mount processing, a TVC is selected as the I/O TVC. All I/O operations associated with the virtual tape drive to which the mount was issued are routed from its vNode to the TVC on the selected cluster.

A Multi Cluster Grid configuration presents a single Composite Library image to the host; the entire subsystem appears as a single tape subsystem to the attached hosts. The host does not see the underlying Distributed Libraries. Multiple TS7700 Virtualization Engine grid configurations can be attached to host systems and operate independently of each another.

### ***Copy Export***

Another function that is supported only on the TS7700 is the *Copy Export* function. It enables the client to move physical volumes with data that is critical for continuing business operations outside a library to an off-site vault.

The expected use is for clients with a Single Cluster Grid who require that data is moved off-site, or for clients with a Multi Cluster Grid where a primary need is validating and testing a client-described disaster scenario.

Copy Export supports moving the logical copy of the original logical volume to another location and keeping the original logical volume within the original TS7700 and available for normal production. A related function is called *Copy Export Recovery*. With this function, you insert the physical volumes and establish the TS7700 again based on the contents of the physical volumes.

### ***Tape encryption***

Encryption is also supported on TS7700. Encryption of backstore tapes helps control the risks of unauthorized data access without an excessive security management burden or subsystem performance issues. IBM tape encryption solutions all use an Encryption Key Manager (EKM) or the Tivoli Key Lifecycle manager (TKLM) as a central point from which all encryption keys are handled. The EKM or TKLM communicates with the TS7700 Virtualization Engine, and Encryption on the TS7700 is controlled outboard on a physical stacked volume pool basis and is defined using the Management Interface (MI). As you set up encryption for all logical volumes in a specific pool, you can use it in addition to the Copy Export Function.

### ***Secure Data Erase function***

Expired data on a physical volume remains readable until the volume has been completely overwritten with new data. This situation might be a concern for clients who want to find an older version of a data volume. Also, retrieving an older data volume can be costly.

Physical volume erasure on a physical volume pool basis is controlled by an additional reclamation policy. With the Secure Data Erase function, all reclaimed physical volumes in that pool are erased by writing a random pattern across the whole tape prior to being reused. In the case of a physical volume that has encrypted data, erasing can involve basically *shredding* the encryption keys on the volume to accomplish the erasure.

### ***Copy Consistency Points***

In a grid environment, you specify where and when you want to have a copy of your data. Currently, three settings are available: These three settings are available:

- ▶ RUN

Copy occurs as part of the rewind-unload operation and before the rewind-unload operation at the host completes. This mode is comparable to the immediate copy mode of the PtP VTS.

- ▶ Deferred

Copy occurs as part of the rewind-unload operation and before the rewind-unload operation at the host completes. This mode is comparable to the deferred copy mode of the PtP VTS.

- ▶ No Copy

No copy is made. For each cluster in a Multi Cluster Grid, you specify a setting for the cluster and a setting for the other clusters.

The settings do not need to be the same on all clusters. When a volume is mounted on a virtual tape device, the Copy Consistency Point policy of the cluster to which the virtual device belongs is honored unless you specify Retain copy mode in which case the initial setting are retained.

## **2.15 Partitioned data sets**

A partitioned data set (PDS) is a data set containing multiple members, each member containing a sequential data set, and a directory to manage the members. This type of data set is often used to hold load modules and source program libraries, such as DB2 for z/OS does. PDS files reside only on disk, not on tape, to use the directory structure to access individual members, not on tape. They are most often used for storing multiple JCL files, utility control statements and executable modules.

Deleting or replacing members requires some maintenance to re-establish space distribution.

An improvement of this structure has been provided by the partitioned data set extended (PDSE) introduced with the MVS/XA system.

In this section, we describe PDS and PDSE and highlight these improvements.

### **2.15.1 PDS**

A PDS is essentially made up of two parts: a directory and members. The directory is a set of contiguous 256-byte blocks, present at the beginning of the data set. Each of these directory

blocks contains a 2-byte count field at the beginning and from 3 to 21 directory entries after that. There is one directory entry for each member in the PDS. Each directory entry contains 8-byte member name (padded with spaces, if needed), starting position of the member in the PDS (in TTR relative track addressing format) and some optional (up to 62 bytes) user data.

A directory block contains only as many complete entries as can fit in 254 bytes (2 bytes are reserved for count field). The remaining bytes are left unused. The length of the user data determines how many complete entries can fit in one directory block. The 2-byte count field contains the number of used (also called *active*) bytes, including the bytes used for the count field.

A PDS is limited to 64K tracks because of the TTR used in the directory. Also, a PDS can have a maximum of 16 extents.

## 2.15.2 PDSE

The limited directory structure was the reason why there was a need to improve the PDS. When the PDSE was introduced, it replaced the rigid directory structure of the PDS with a new flexible scheme and also brought in many other new features, while keeping the PDSE backward compatible with PDS. Except for low-level (hardware-dependent) processing, users of PDS do not have to be aware of the change to PDSE.

A PDSE consists of a directory and zero or more members, like a PDS. But there are differences between a PDS and a PDSE.

A PDSE can be created with JCL, TSO/E, and Interactive System Productivity Facility (ISPF), just like a PDS, and can be processed with the same access methods. PDSE data sets are stored only on DASD, not on tape.

The directory can expand automatically as needed, up to the addressing limit of 522,236 members. It also has an index, which provides a fast search for member names. Space from deleted or moved members is automatically reused for new members, so you do not have to compress a PDSE to remove wasted space. Each member of a PDSE can have up to 15,728,639 records. A PDSE can have a maximum of 123 extents, but it cannot extend beyond one volume. When a directory of a PDSE is in use, it is kept in processor storage for fast access.

PDSE data sets can be used in place of nearly all PDS data sets that are used to store data. However, the PDSE format is not intended as a PDS replacement. When a PDSE is used to store load modules, it stores them in structures called *program objects*.

A PDSE is similar to a PDS. Each member name can be eight bytes long. For accessing a PDS directory or member, most PDSE interfaces are indistinguishable from PDS interfaces. PDS and PDSE data sets are processed using the same access methods (for example, BSAM, QSAM and BPAM). Within a given PDS or PDSE, the members must use the same access method.

However, PDSE data sets have a different internal format, which gives them increased usability. You can use a PDSE in place of a PDS to store data or programs. In a PDS, you store programs as load modules. In a PDSE, you store programs as program objects. If you want to store a load module in a PDSE, you must first convert it into a program object (using the IEBCOPY utility).

PDSE data sets have several features that can improve user productivity and system performance. The main advantage of using a PDSE over a PDS is that a PDSE automatically reuses space within the data set without the need for anyone to periodically run a utility to

reorganize it. The system reclaims space automatically when a member is deleted or replaced, and returns it to the pool of space available for allocation to other members of the same PDSE. The space can be reused without having to do an IEBCOPY compress.

Also, the size of a PDS directory is fixed regardless of the number of members in it; the size of a PDSE directory is flexible and expands to fit the members stored in it.

Other advantages of PDSE data sets are as follows:

- ▶ PDSE members can be shared. This characteristic makes maintaining the integrity of the PDSE easier when modifying separate members of the PDSE at the same time.
- ▶ The system requires less time to search a PDSE directory. The PDSE directory, which is indexed, is searched using that index. The PDS directory, which is organized alphabetically, is searched sequentially. The system might cache storage directories of frequently used PDSE data sets.
- ▶ You may create multiple PDSE members at the same time. For example, you can open two data control blocks (DCBs) to the same PDSE and write two members at the same time.
- ▶ PDSE data sets contain up to 123 extents. An *extent* is a continuous area of space on a DASD storage volume, occupied by or reserved for a specific data set.
- ▶ When written to DASD, logical records are extracted from the user's blocks and are reblocked. When read, records in a PDSE are reblocked into the block size specified in the DCB. The block size used for the reblocking can differ from the original block size.

## 2.16 New technologies and functions benefit DB2 for z/OS

This section lists the storage hardware technology and z/OS enhancements that have been made available during the last decade to deliver new capabilities.

Both the new hardware functions and z/OS enhancements bring benefits to DB2 for z/OS.

The following list indicates many of the hardware functions and z/OS enhancements:

- ▶ DS8000 striping and access method striping
- ▶ RAID
- ▶ Large volume support
- ▶ PAV, HyperPAV, and Dynamic PAV
- ▶ Multiple allegiance
- ▶ Priority queuing
- ▶ Extended address volume (EAV)
- ▶ Dynamic volume expansion
- ▶ New DS8000 cache algorithm
- ▶ MIDAW
- ▶ zHPF
- ▶ Solid-state drives (SSD)
- ▶ Easy Tier
- ▶ DS8000 Copy Services
- ▶ Compression
- ▶ Tape virtualization





## DB2 storage objects

In this chapter, we provide an introduction to DB2 for z/OS (DB2 throughout this book) for the storage administrators interested in understanding various types of data-related objects that are used in a DB2 environment. Special emphasis is placed on the data sets that are managed directly by DB2.

This chapter contains the following topics:

- ▶ DB2 overview
- ▶ Non-VSAM data sets
- ▶ DB2 data objects
- ▶ Creating table spaces and index spaces
- ▶ Large sequential data sets
- ▶ DB2 system table spaces
- ▶ DB2 application table spaces
- ▶ Index compression
- ▶ DB2 recovery data sets
- ▶ DFSMSHsm and DB2 BACKUP SYSTEM
- ▶ RESTORE SYSTEM utility
- ▶ Online CHECK DATA
- ▶ Other DB2 data sets
- ▶ DB2 data sets naming conventions
- ▶ Format and preformat of DB2 data sets

## 3.1 DB2 overview

DB2 is a database management system that was originally based on the relational data model and is now extended to include hybrid object-relational and XML models. Many customers use DB2 for applications that require good performance or high availability for large amounts of data. This data is stored in data sets that are directly associated to DB2 table spaces and distributed across DB2 databases. Data in table spaces is often accessed through indexes; indexes are stored in index spaces.

Data table spaces can be divided into two groups: system table spaces and user table spaces. Both groups have identical data attributes. The difference is that system table spaces are required to control and manage the DB2 subsystem and the user data. The consequence of this is that system table spaces require the highest availability and some special consideration. User data cannot be accessed without system data or with obsolete system data.

In addition to the data table spaces, DB2 requires a group of traditional data sets, not associated to table spaces, that are used by DB2 to help provide the appropriate high level of data availability: the back-up and recovery data sets. Proper management of these data sets is required to achieve this objective.

In summary, the main data set types in a DB2 subsystem are:

- ▶ DB2 back-up and recovery data sets
- ▶ DB2 system table spaces and index spaces
- ▶ DB2 user table spaces and index spaces

## 3.2 Non-VSAM data sets

Typical non-VSAM data sets in DB2 environments are data sets such as partitioned data sets for the DB2 modules, and sequential data sets for image copy utility output and utility work areas.

The DB2 SDSNLOAD data set contains most of the DB2 executable code. It is allocated to a partitioned data set (PDS). In DB2 V8, you had the option to convert this data set from PDS to a partitioned data set extended (PDSE). If you did not take the opportunity during your DB2 V8 migration to convert this data set to a PDSE, you must now do so because in DB2 9 for z/OS, SDSNLOAD must now be allocated (with the DSNALLOC job) as a PDSE data set, mainly to help overcome the space limitations of the PDS.

Also, be aware of the operational differences between PDS and PDSE data sets. These are explained in 2.15, “Partitioned data sets” on page 111 and detailed in the *DB2 Program Directory* that is included with the DB2 product.

**Note:** SMP/E must “know” which type of data set it is managing. It compresses a PDS, but not compress a PDSE. If you change the data set organization, you must also update the SMP/E definition

## 3.3 DB2 data objects

DB2 manages data by associating it to a set of the following DB2 objects:

- ▶ Table
- ▶ Table space
- ▶ Index
- ▶ Index space
- ▶ Database
- ▶ Storage group
- ▶ Alias
- ▶ Synonym
- ▶ View

Several of these objects (table space, index space, storage group) map to data sets or volumes and have a physical representation on storage devices. Other objects (table, index, alias, synonym, view) are application-oriented, not directly mapped to space on disk. A DB2 database is a DB2 structure representing a collection of related objects.

A complete description of all DB2 objects and their implementation is in the *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840, implementing and altering database design.

### 3.3.1 Table

All data managed by DB2 is associated to a table. A *table* is a set of data elements (values) organized using a model of columns (identified by their name) and horizontal rows. A table has a specified number of columns, but can have any number of rows. The data within the table represents the unit of data identified and addressed by the user.

The table is the main object used by DB2 applications. The SQL DML used by application programs and users directly references data in tables. The types of tables are as follows:

- ▶ Base tables
- ▶ Temporary tables, created as global or declared global
- ▶ Auxiliary tables, for storing LOB and XML data
- ▶ Clone tables, of existing base tables
- ▶ Materialized query tables, defined by the result of a query

When defining a table, the maximum number of columns in a table can affect the page size. The page size affects the size of the table space. Table 3-1 shows the maximum number of columns and maximum number of characters in a row (row length).

*Table 3-1 Maximum column and row length per page for tables*

	4 KB page	8 KB page	16 KB page	32 KB page
<b>Maximum columns</b>	750	750	750	750
<b>Maximum row length</b>	4058	8138	16330	32714

When an EDITPROC is defined on a table, the maximum row size is 10 bytes less.

### 3.3.2 Table space

A table space is used to store one or more tables. A *table space* is physically implemented with one or more data sets. Table spaces are VSAM linear data sets (LDS). Because table spaces can be larger than the largest possible VSAM data set, a DB2 table space might require more than one VSAM linear data set.

The page is the unit of storage within a table space (or index space) and its size must match the size of the buffer in the buffer pool used by DB2 for accessing data. Pages can be 4 KB, 8 KB, 16 KB, or 32 KB in size. In a table space, a page contains one or more rows of a table. A page corresponds to a VSAM Control Interval if the DSNZPARM DSVCI=YES (the default) is specified or a chain (2, 4 or 8) of 4 KB CIs managed by DB2.

These DB2 VSAM LDS data sets, for table space and index spaces, are defined as follows:

- ▶ DB2 managed or user managed:
  - DB2 managed data sets are created using Data Definition Language (DDL).
  - User managed data sets are created outside of DB2, typically through IDCAMS.
- ▶ SMS or non-SMS managed:
  - For SMS data sets, the LDS can be defined by using the DB2 STOGROUP, SMS storage group, or both.

**Note:** We suggest that when using SMS and DB2 STOGROUPS to manage your DB2 table space sets, use VOLUMES (“\*”) and let SMS manage this for you.

Introduced in DB2 9, you can specify the SMS DATACLAS, STORCLAS, or MGMTCLAS attributes. You then do not have to specify the VOLUMES attribute.

DB2 provides the following types of table spaces:

- ▶ Simple: Contains one or more tables. You cannot create *new* tables (as of DB2 9); this table space will be replaced by a universal table space called partition-by-growth.
- ▶ Segmented: Contains one or more tables.
- ▶ Partitioned: Contains one, often large, table. This type is being deprecated and replaced by universal table space partition-by-range.
- ▶ LOB: Holds large object data.
- ▶ Universal: Combines partitioned and segmented characteristics, and can contain both a base and a clone table.
- ▶ XML: Contains an XML column.

#### Physical sizes and limits of table spaces

This section lists the characteristics of physical sizes by table space type.

##### ***Simple table spaces***

You can have single and multiple tables for each table space, and all rows from all tables can share the same page in that table space. Simple table spaces are still supported in DB2 9 but you cannot create new ones. A simple table space has a maximum size of 64 GB.

##### ***Segmented table spaces***

Segmented table spaces allows you to create single or multiple tables in a single table space. Each table has its own segments. You pick the segment size in the range of 4 - 64 pages in multiples of four. A segmented table space has a maximum size of 64 GB.

For both simple and segmented table spaces, the maximum 64 GB size corresponds to the concatenation of up to 32 data sets, each of up to 2 GB in size.

### ***Partitioned table spaces***

A partitioned table space divides the data into separate data sets known as partitions. A partition table space contains only one table. Each partition can have different characteristics, such as volume placement, compression, and free space. You can have up to 4,096 partitions of up to 64 GB each. The total table size is limited to 16 TB for a 4 KB page size. The number of partitions depends on the value specified on the DSSIZE parameter. (See “DSSIZE” on page 120 for details.)

### ***LOB table spaces***

Large object (LOB) table spaces are used to hold large object data. LOB table spaces are associated with the table space that holds the logical LOB column. LOB table spaces can be explicitly defined. This table space must exist in the same database as the base table, and it will contain the auxiliary table. You can have 254 partitions of 64 GB each, allowing a total of 16 TB, and there will be one LOB table space for each LOB column (up to 254 partitions).

Because each data set of a LOB table space can grow up to 64 GB, and there can be up to 254 data sets per table space, the maximum size of a non-partitioned LOB table space is 16,256 GB (16 TB) in total. Because the number of partitions can grow up to 4,096 in the base table, there are 4,096 single LOB table spaces, each holding up to 16 TB of data as a maximum size. This gives a grand total of  $4,096 \times 16 \text{ TB} = 65,536 \text{ TB}$  available for a single column of LOB data.

Multimedia application environments rely on many types of large data objects. Those objects can be large text documents, X-ray images, audio messages, pictures, and many other types of images.

The data types provided by DB2, such as VARCHAR or VARBINARY (DB2 9), are not large enough to hold a large amount of data because of their limit of 32 KB. Support for large data objects (LOBs) is based on the set of data types that were introduced with DB2 V6.

DB2 provides LOBs that support strings of up to 2 GB in size, well beyond the 32 KB that are supported by a varchar column. Techniques for storing and retrieving these huge amounts of data have also been created within DB2. The LOB data types allow you to store directly in DB2 tables objects in size of up to 2 GB, and 65,536 TB per LOB column. Their characteristics depend on the way they are stored in your DB2 subsystem:

- ▶ Character large objects (CLOBs)
- ▶ Binary large objects (BLOBs)
- ▶ Double-byte character large objects (DBCLOBs)
- ▶ For its structure support, DB2 uses the ROWID data type

### ***Universal table spaces***

DB2 9 for z/OS introduced the universal table space (UTS). This kind of table space is *both* segmented and partitioned. Two types of universal table spaces are available:

- ▶ PBG: Partition-by-growth table space
- ▶ PBR: Partition-by-range or range-partitioned table space

The major benefit regarding UTS is that you can get a segmented table space that can grow up to 128 TB of data (we assume you defined the correct DSSIZE and the correct number of partitions are specified). The addition of performance of a mass DELETE (previously available only for segmented table spaces) for a partitioned structure and the support for CLONE tables are the key functions of UTS.

## ***XML table spaces***

An XML table space, such as a LOB table space, is a table space which contains the auxiliary table. If an XML column is defined, the rows containing the XML data are contained in a separate table and table space from the base table. DB2 implicitly creates an XML table space (UTS) and an XML table to store the XML data, along with a node identifier (ID).

Each XML column has its own table space. The XML table space does not have limit keys. The XML data resides in a partition that corresponds in number to the partition number of the base row.

XML table spaces share or inherits the CCSID, DSSIZE, LOCKMAX and LOG attributes from the base table space.

## **Table space attributes that affect space**

On the CREATE statement for the table space are parameters that influence the size and other attributes of the underlying DB2 VSAM linear data sets. This section describes these attributes and how they affect the definition.

### ***DSSIZE***

This parameter specifies the maximum size of any partition and of any LOB data set. This parameter controls the *maximum* size a data set can grow to and has no correlation to the PRIQTY or SECQTY values. All UTS spaces have a record ID (RID) length of 5 bytes, regardless of the DSSIZE value. For non-UTS, the RID is always 4 bytes for DSSIZE less than 4 GB. We mention this information because it affects the space usage of the table space.

The maximum number of partitions of a table space depends on the page size and DSSIZE. The total table space size depends on the number of partitions and DSSIZE.

Page size affects table size because it affects how many partitions a table space can have. Table 3-2 shows the maximum number of partitions for DSSIZE and page size and total table space size for data sets that are both *enabled* or *not enabled* for extended addressability (EA). Specifying a DSSIZE greater than 4 GB requires EA-enabled data sets.

*Table 3-2 Table space sizes and total space sizes*

<b>RID Type</b>	<b>Page size</b>	<b>DSSIZE</b>	<b>Max # of parts</b>	<b>Total TS sizes</b>
5 Byte EA	4 KB	1 GB	4096	4 TB
5 Byte EA	4 KB	2 GB	4096	8 TB
5 Byte EA	4 KB	4 GB	4096	16 TB
5 Byte EA	4 KB	8 GB	2048	16 TB
5 Byte EA	4 KB	16 GB	1024	16 TB
5 Byte EA	4 KB	32 GB	512	16 TB
5 Byte EA	4 KB	64 GB	256	16 TB
<b>5 Byte EA</b>	<b>8 KB</b>	<b>1 GB</b>	<b>4096</b>	<b>4 TB</b>
<b>5 Byte EA</b>	<b>8 KB</b>	<b>2 GB</b>	<b>4096</b>	<b>8 TB</b>
<b>5 Byte EA</b>	<b>8 KB</b>	<b>4 GB</b>	<b>4096</b>	<b>16 TB</b>
<b>5 Byte EA</b>	<b>8 KB</b>	<b>8 GB</b>	<b>4096</b>	<b>32 TB</b>
<b>5 Byte EA</b>	<b>8 KB</b>	<b>16 GB</b>	<b>2048</b>	<b>32 TB</b>

RID Type	Page size	DSSIZE	Max # of parts	Total TS sizes
<b>5 Byte EA</b>	<b>8 KB</b>	<b>32 GB</b>	<b>1024</b>	<b>32 TB</b>
<b>5 Byte EA</b>	<b>8 KB</b>	<b>64 GB</b>	<b>512</b>	<b>32 TB</b>
5 Byte EA	16 KB	1 GB	4096	4 TB
5 Byte EA	16 KB	2 GB	4096	8 TB
5 Byte EA	16 KB	4 GB	4096	16 TB
5 Byte EA	16 KB	8 GB	4096	32 TB
5 Byte EA	16 KB	16 GB	4096	64 TB
5 Byte EA	16 KB	32 GB	2048	64 TB
5 Byte EA	16 KB	64 GB	1024	64 TB
<b>5 Byte EA</b>	<b>32 KB</b>	<b>1 GB</b>	<b>4096</b>	<b>4 TB</b>
<b>5 Byte EA</b>	<b>32 KB</b>	<b>2 GB</b>	<b>4096</b>	<b>8 TB</b>
<b>5 Byte EA</b>	<b>32 KB</b>	<b>4 GB</b>	<b>4096</b>	<b>16 TB</b>
<b>5 Byte EA</b>	<b>32 KB</b>	<b>8 GB</b>	<b>4096</b>	<b>32 TB</b>
<b>5 Byte EA</b>	<b>32 KB</b>	<b>16 GB</b>	<b>4096</b>	<b>64 TB</b>
<b>5 Byte EA</b>	<b>32 KB</b>	<b>32 GB</b>	<b>4096</b>	<b>128 TB</b>
<b>5 Byte EA</b>	<b>32 KB</b>	<b>64 GB</b>	<b>2048</b>	<b>128 TB</b>
5 Byte non-EA large	4 KB	4 GB	4096	16 TB

### ***PRIQTY and SECQTY***

When using DB2 managed data sets, you can specify the space with DDL by using PRIQTY and SECQTY attributes:

- PRIQTY and SECQTY are specified in KB.

Allocations are slightly higher when specifying DSVCI=YES<sup>1</sup> and you are allocating a 32 KB page (because VSAM uses physical blocks of 16 KB to better fit the track):

- 1 track = 48 KB
- 1 cylinder = 720 KB (15 tracks multiplied by 48 KB per track); if the allocation is above 672 tracks, it is allocated in cylinders because of the CA boundary
- If PRIQTY>1 cylinder and SECQTY<1 cylinder, secondary rounded up to 1 cylinder (CA 1 cylinder)
- If PRIQTY<1 cylinder and SECQTY>1 cylinder, allocations are in the tracks, not in the cylinders (CA<1 cylinder)
- See APAR PK05644: preformat up to 2 cylinders at a time regardless of track or cylinder allocation
- Preformat starting in DB2 9 is up to 16 cylinders at a time, up from 2 cylinders, if the allocation is 16 cylinders.

<sup>1</sup> DSVCI is field 5 of installation panel DSNTIP7, better known as VARY DS CONTROL INTERVAL.

- ▶ A track can hold 56 KB, however VSAM allocates and uses only 48 KB for LDS. For one track, specify a PRIQTY of 48 KB, and for one cylinder you specify 720 KB (which is 48 KB multiplied by 15 tracks).
- ▶ If SECQTY is not specified, the value is chosen through a combination of sliding secondary allocation and DSNZPARM values for TSQTY, IXQTY
- ▶ If SECQTY is specified with a value of 0, then no secondary is allowed.

DB2 V8 introduced support for VSAM CI sizes of 8, 16, and 32 KB. These sizes are enabled by the DSNZPARM parameter, *DSVCI*, option 6 on panel DSNTIP7. This parameter is valid for both user-defined and DB2 STOGROUP-defined table spaces.

After DSVCI is set to YES, the VSAM control interval (CI) size will be the same as the page size and the VSAM physical block size.

A 32 KB page is the exception for good reason. If you have a page size of 8 KB, you have a CI size of 8 KB and a physical block size of 8 KB; a page size of 16 KB has a CI size of 16 KB and a physical block size of 16 KB. However if you have a page size of 32 KB with a 32 KB CI, the CI spans two physical 16 KB blocks. The 32 KB exception is there for space utilization. With a 48 KB track size, only one 32 KB CI can fit per track if a 32 KB block is used. Using 16 KB blocks allows VSAM to span tracks for 32 KB CI and VSAM handles all of the I/O.

With DB2 9, index spaces can use 4, 8, 16, and 32 KB pages. After you have activated the new CI sizes (in DB2 V8 NFM), all new index and table spaces are allocated by DB2 with a CI corresponding to the page size.

When using DB2 user-defined objects (USING VCAT), it is your responsibility to define the underlying VSAM cluster with the correct CISE. As mentioned previously, DB2 uses VSAM LDS for table space and indexes. LDS only use CI sizes that are a multiple of 4 KB and no control interval definition fields (CIDs). VSAM writes the CIs in physical blocks for the best fit within the track.

For a 32 KB page size allocation, using a new CI size, note the following information:

- ▶ Because a 16 KB block is not used, every fifteenth track of a cylinder (one cylinder CA size) for 32 KB pages, you lose 2.2% of each cylinder.
- ▶ For DB2-managed data sets, DB2 adds 2.2% back into the allocation.

For example, if you are creating a 32 KB table space with a PRIQTY of 72000 (which is the equivalent of 100 cylinders), with DSVCI=YES setting, your allocation is 103 cylinders. DB2 recognizes that you have requested a 32 KB page with the new CI size and automatically adds the 2.2% overhead to your allocation that you are losing because of the new mechanism. See Example 3-1.

*Example 3-1 DDL of with DSVCI enabled*

---

```
CREATE TABLESPACE TSHM32K2
  IN DSNHMDBE
  USING STOGROUP STOGFEA
  PRIQTY 72000 SECQTY 1440
  FREEPAGE 3 PCTFREE 5
  GBPCACHE CHANGED
  TRACKMOD YES
  Numparts 4
  BUFFERPOOL BP32K
  LOCKSIZE ANY
  LOCKMAX SYSTEM
  CLOSE NO
```

---



Because a one-cylinder CA is used in this example, the 102.2 cylinders is rounded up to 103 cylinders. See Example 3-2.

*Example 3-2 VSAM LISTCAT output of TS with DSVCI enabled*

---

```

CLUSTER ----- DB9AU.DSNDBC.DSNHMDBE.TSHM32K2.I0001.A001
  IN-CAT --- UCAT.DB9A
  HISTORY
    DATASET-OWNER----PAOLOR9      CREATION-----2010.123
    RELEASE-----2              EXPIRATION-----0000.000
  SMSDATA
    STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
    DATACLASS  -----DB2EFEA    LBACKUP ---0000.000.0000
    EATTR----- (NULL)
  ...

VOLUME
  VOLSER-----SBOXOU      PHYREC-SIZE-----16384
  DEVTYPE-----X'3010200F'  PHYRECS/TRK-----3
  VOLFLAG-----PRIME      TRACKS/CA-----15

ALLOCATION
  SPACE-TYPE-----CYLINDER    HI-A-RBA-----74252288
  SPACE-PRI-----103          HI-U-RBA-----11796480
  SPACE-SEC-----3

EXTENTS:
LOW-CCHH-----X'04F90000'    LOW-RBA-----0      TRACKS-----1545
HIGH-CCHH----X'055F000E'     HIGH-RBA-----74252287

```

---

If you are creating a 32 KB table space with a PRIQTY of 72000 with explicit DSVCI=NO set, the DDL is as shown in Example 3-3.

*Example 3-3 DDL with DSVCI disabled*

---

```

CREATE TABLESPACE TSHM32K0
  IN DSNHMDBE
  USING STOGROUP STOGFEFA
  PRIQTY 72000 SECQTY 1440
  FREEPAGE 3 PCTFREE 5
  GBPCACHE CHANGED
  TRACKMOD YES
  DSSIZE 1 G
  NUMPARTS 2
  BUFFERPOOL BP32K
  ...

```

---

In this case the allocation is 100 cylinders as shown in Example 3-4 on page 124.

Example 3-4 IDCAMS LISTCAT output of TS with DSVCI disabled

---

```

CLUSTER ----- DB9AU.DSNDBC.DSNHMDBE.TSHM32K0.I0001.A001
  IN-CAT --- UCAT.DB9A
  HISTORY
    DATASET-OWNER----PAOLOR9      CREATION-----2010.123
    RELEASE-----2              EXPIRATION-----0000.000
  SMSDATA
    STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
    DATACLASS  -----DB2EFEA    LBACKUP ---0000.000.0000
    EATTR----- (NULL)
  ...

VOLUME
  VOLSER-----SBOX1W      PHYREC-SIZE-----4096
  DEVTYPE-----X'3010200F'  PHYRECS/TRK-----12
  VOLFLAG-----PRIME      TRACKS/CA-----15
LOCATION
SPACE-TYPE-----CYLINDER    HI-A-RBA-----73728000
SPACE-PRI-----100        HI-U-RBA-----11796480
SPACE-SEC-----2
EXTENTS:
LOW-CCHH-----X'02070000'   LOW-RBA-----0      TRACKS-----1500
HIGH-CCHH-----X'026A000E'  HIGH-RBA-----73727999

```

---

### SECQTY

This SECQTY value specifies the minimum secondary space allocation for a DB2-managed data set. DB2 space allocations sliding scale for secondary extents is enabled by setting the DSNZPARM MGEXTSZ to YES and an object's SECQTY to -1 (minus one). See 3.13.7, “DSNZPARMs affecting DB2 data set sizes” on page 176.

## 3.3.3 Index

A table can have zero or more indexes. An index contains keys. Each key may point to one or more data rows. The purpose of an *index* is to establish a way to get a direct and faster access to the data in a table. An index with the UNIQUE attribute enforces distinct keys and uniqueness of all rows in the referenced table. An index with the CLUSTER attribute can be used to establish and maintain a physical sequence in the data.

Prior to DB2 9, the size of an index page was limited to 4 KB. The size of an index page limits the number of index keys that the index page can accommodate and can cause contention in indexes that split frequently. DB2 9 lifts these restrictions by offering expanded index page sizes of 8 KB, 16 KB, and 32 KB. An index page size that is greater than 4 KB accommodates more index keys per page and can reduce the frequency of index page splits. You can use the INDEXBP option on both CREATE DATABASE and ALTER DATABASE statements to specify 4 KB, 8 KB, 16 KB, or 32 KB index buffer pools. You can also use the BUFFERPOOL keyword on the CREATE INDEX statement to specify 8 KB, 16 KB, and 32 KB buffer pools.

The larger page sizes are required for the use of compression of indexes.

An asymmetrical split of index pages is an enhancement, introduced in DB2 9, and that has the potential to allow index sizes to be reduced and to allow for more densely packed indexes.

Other scalability implications that this may bring are as follows:

- ▶ Index page split latch contention relief
- ▶ Larger index page
- ▶ Fewer index splits
- ▶ Latch contention reduction, especially in data sharing environments

When all entries of an index leaf page are consumed during inserts, page-split processing takes place and DB2 allocates a new page and moves some entries from the old page to the new page. During page-split processing, DB2 locks the entire index tree and blocks every load job, except one, from processing. This can reduce data load rate significantly. Read the section about “Relief for sequential key insert” in *DB2 9 for z/OS Performance Topics*, SG24-7473 for a detailed discussion of how larger page size, asymmetric split, and the RANDOM parameter can help reduce index split processing overhead.

### 3.3.4 Index space

An *index space* is used to store an index. An index space is physically represented by one or more VSAM LDS data sets.

When a non-partitioning index must be split across multiple data sets to help improve I/O performance, these particular types of data sets are called PIECEs. you can have up to 254 pieces each of up to 64 GB in size.

### 3.3.5 Database

A *database* is a DB2 representation of a group of related objects. Each of the previously named objects has to belong to a database. DB2 databases are used to organize and manage these objects. Normally a database is the association of table spaces and index spaces used by an application or a coherent part of an application.

### 3.3.6 Storage group

A DB2 *storage group* (STOGROUP) is a list of storage volumes. STOGROUPs are assigned to databases, table spaces, or index spaces when using DB2-managed objects. DB2 uses STOGROUPs for disk allocation of the table and index spaces.

Installations that are SMS-managed can define storage groups in DB2 9 for z/OS with the SMS attributes as shown by the syntax diagram in Figure 3-1.

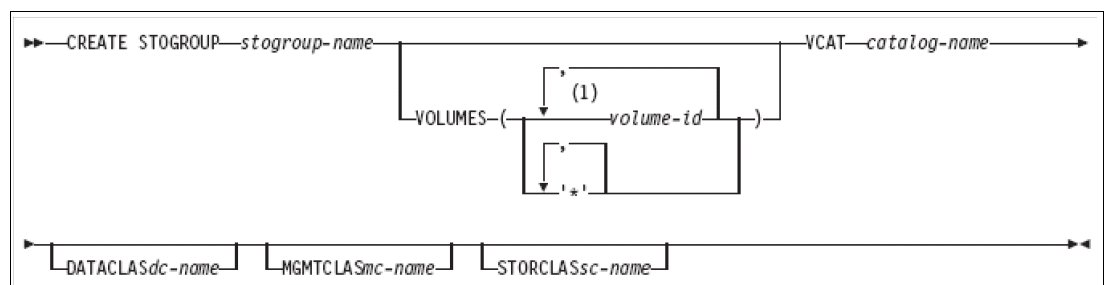


Figure 3-1 CREATE STOGROUP syntax diagram with SMS attributes

When processing the VOLUMES, DATACLASS, MGMTCLASS, or STORCLASS clauses, DB2 does not check the existence of the volumes or classes or determine the types of devices that

are identified or if SMS is active. Later, when the storage group allocates data sets, the list of volumes is passed in the specified order to Data Facilities (DFSMSdfp).

See Example 3-5 for DDL for STOGROUP creation with DATA CLASS specified.

*Example 3-5 DDL for STOGROUP creation with DATA CLASS specified*

---

```
CREATE STOGROUP STOGEFEA
      VCAT DB9AU DATACLAS DB2EFEA;
```

---

In Example 3-6, we show the DB2 catalog information from SYSIBM.SYSSTOGROUP table for the STOGROUP created previously with the DATACLAS attribute.

*Example 3-6 DB2 catalog information for a STOGROUP*

---

```
DB2 Admin ----- DB9A Interpretation of an Object in SYSSTOGROUP --
Option ==>
```

Details for storage group : **STOGEFEA**

```
Storage group owner      : DB2R2
Created by               : DB2R1
Created timestamp       : 2010-05-03-13.12.59.281472
Altered timestamp       : 2010-05-03-13.12.59.281472
Creator type            : Auth ID
VSAM catalog name      : DB9AU
DASD space allocated (KB) : 1440 (Float: 1.4400000000000000E+03)
Last update of space field : 10123 (yyddd)
STOSPACE last executed  : 2010-05-03-15.54.41.049576
SMS Data class       : DB2EFEA
SMS Management class    :
SMS Storage class       :
Created in DB2 Version   : M - DB2 V9
```

---

Table 3-3 denotes the difference between DB2 STOGROUPs and SMS storage groups.

*Table 3-3 DB2 STOGROUPs and SMS storage groups differences*

DB2 STOGROUP	SMS storage group
Separate STOGROUPS share the same DASD volumes.	A disk volume can only belong to <b>one</b> SMS storage group
The VOLSERS are specific.	Consider coding VOLUMES("(*)"). This coding enable SMS management. Consider avoiding Guaranteed Space and specific VOLSERS where possible. They defeat the purpose of SMS.
SYSIBM.SYSVOLUMES contains one row for each volume in the VOLID column.	SYSIBM.SYSVOLUMES contains an "(*)" for each volume in the VOLID column. This is true when your specified VOLUMES("(*)").
Limited to manage up to 133 volumes.	No limit is specified.
Volume selection is based on free space.	Volume selection is based on SMS algorithms.

## 3.4 Creating table spaces and index spaces

Table and index spaces can be created in either of the following ways:

- ▶ DB2-defined and managed
- ▶ User-defined and managed

For a detailed explanation on this subject, see “Designing Storage Groups and Managing DB2 Data Sets” in *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840.

DB2-defined and managed spaces should be the choice by default. It is the easier of these solutions and is adequate for most table and index spaces in the majority of situations.

User-defined table spaces provide more control of the data set placement and all VSAM definition options are available. Examples of where user defined table spaces might be required are as follows:

- ▶ Table spaces that require specific DFSMS classes
- ▶ Table spaces with critical performance and availability requirements (such as DB2 catalog and directory)

### 3.4.1 DB2-defined and managed

Example 3-7 shows a DB2-defined table space. The CREATE TABLESPACE resolves the physical allocation and defines this table space to DB2. The STOGROUP SYSDEFLT defines a set of volumes for the data set and PRIQTY specifies the size in kilobytes (KB).

*Example 3-7 CREATE TABLESPACE DDL: DB2-defined*

---

```
CREATE TABLESPACE PAOLOR1
  IN DSNHMDDB
  USING STOGROUP SYSDEFLT
  PRIQTY 1440 SECQTY 720
  ERASE NO
  FREEPAGE 3 PCTFREE 5
  GBPCACHE CHANGED
  TRACKMOD YES
  SEGSIZE 4
  BUFFERPOOL BP1
  LOCKSIZE ANY
  LOCKMAX SYSTEM
  CLOSE NO
  COMPRESS NO
  CCSID EBCDIC
  DEFINE YES
  MAXROWS 255;
COMMIT;
```

---

Example 3-8 shows the corresponding generated VSAM clusters.

*Example 3-8 IDCAMS LISTCAT output of a DB2 defined TS*

---

```

CLUSTER ----- DB9AU.DSNDBC.DSNHMDB.PAOLOR1.I0001.A001
  IN-CAT --- UCAT.DB9A
  HISTORY
    DATASET-OWNER----PAOLOR9      CREATION-----2010.120
    RELEASE-----2              EXPIRATION-----0000.000
  SMSDATA
    STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
    DATACLASS  -----(NULL)      LBACKUP ---0000.000.0000
    EATTR----- (NULL)
...

```

---

Indexes are created with a CREATE INDEX statement. The CREATE INDEX statement defines both the index and the associated index space. The CREATE INDEX statement also loads the index with entries that point to the data (index rows) unless the DEFER YES parameter is specified.

### 3.4.2 User defined and managed

Two steps are required to create user defined table spaces or index spaces.

1. The physical allocation is done with an IDCAMS DEFINE CLUSTER; this is shown in Example 3-9.

*Example 3-9 User-defined table space: Step 1, define the cluster*

---

```

DEFINE CLUSTER -
  ( NAME(DB2V910Z.DSNDBC.DSN8D61A.PAOLOR1.I0001.A001) -
    LINEAR -
    REUSE -
    VOLUMES(SBOX10) -
    RECORDS(4096 50) -
    SHAREOPTIONS(3 3) ) -
  DATA -
  ( NAME(DB2V910Z.DSNDBD.DSN8D61A.PAOLOR1.I0001.A001) ) -
  CATALOG(DS2V9)

```

---

2. The table space or index space must be defined to DB2. An example is shown in Example 3-10. It must be noted that the high-level qualifier, the database name, and the table space name in Example 3-9 must match the definitions on Example 3-10.

*Example 3-10 User-defined table space: Step 2, define the table space*

---

```

CREATE TABLESPACE PAOLOR1 IN DSN8D61A
  BUFFERPOOL BPO
  CLOSE NO
  USING VCAT DB2V910Z;

```

---

### 3.4.3 DB2 table spaces on EAV

Example 3-11 on page 129 shows how an EAV is used for DB2 table spaces. The test scenario is of 4 KB page table spaces that are allocated to volume GKDD65. It starts with the allocation and ends with DB2 inserts.

### Example 3-11 Sequence for allocating table spaces in EAV

1. TABLESPACE JOHNNITS1 PRIQTY 7200 (10 cyl) SECQTY 720 (1 cyl)
2. TABLESPACE JOHNNITSL PRIQTY 16194304 (22493 cyl) SECQTY 50 (1 cyl) DSSIZE 64G MAXPARTITIONS 256
3. TABLESPACE JOHNNITS2 PRIQTY 14440 (21 cyl) SECQTY 720 (1 cyl)
4. TABLESPACE JOHNNITS3 PRIQTY 13000 (19 cyl) SECQTY 720 (1 cyl)
5. TABLESPACE JOHNNITSA PRIQTY 9500000 (13195 cyl) SECQTY 50 (1 cyl) DSSIZE 64G MAXPARTITIONS 256
6. TABLESPACE JOHNNITS4 PRIQTY 15120 (21 cyl) SECQTY 720 (1 cyl)
7. TABLESPACE JOHNNITSB PRIQTY 16194304 (22493 cyl) SECQTY 50 (1 cyl) DSSIZE 64G MAXPARTITIONS 256
8. TABLESPACE JOHNNITSC PRIQTY 16194304 (22493 cyl) SECQTY 50 (1 cyl) DSSIZE 64G MAXPARTITIONS 256
9. TABLESPACE JOHNNITSD PRIQTY 16194304 (22493 cyl) SECQTY 50 (1 cyl) DSSIZE 64G MAXPARTITIONS 256
10. TABLESPACE JOHNNITSE PRIQTY 9194304 (12770 cyl) SECQTY 50 (1 cyl) DSSIZE 64G MAXPARTITIONS 256
11. TABLESPACE JOHNNITSF PRIQTY 2983680 (4144 cyl) SECQTY 50 (1 cyl) DSSIZE 64G MAXPARTITIONS 256
12. DROP TABLESPACE JOHNNITS4
13. TABLESPACE JOHNNITS5 PRIQTY 48 (1 track) SECQTY 720 (1 cyl)
14. CREATE TABLE in JOHNNITS5 and insert
15. DROP TABLESPACE JOHNNITS3
16. CREATE TABLE in JOHNNITS2 and insert

Figure 3-2 contains the output of IEHLIST, ISMF, and ISPF about the allocate EAV. It displays the cylinder and track managed space.

## Empty volume GKDD65

**IEHLIST output:**

CONTENTS OF VTOC ON VOL **GKDD65** <THIS IS AN SMS MANAGED VOLUME>  
 NUMBER OF MULTICYLINDER UNITS

CYLINDERS	FIRST CYL ADDR	SPACE
120204	65520	21

THERE ARE 120187 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS ON THIS VOLUME  
 THERE ARE 65503 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS FROM THE TRACK-MANAGED SPACE  
 THERE ARE 8996 BLANK DSCBS IN THE VTOC ON THIS VOLUME  
 THERE ARE 339 UNALLOCATED VIRS IN THE INDEX

**ISMF output:**

VOLUME	FREE SPACE	% FREE	ALLOC SPACE	FRAG INDEX	LARGEST EXTENT	FREE EXTENTS
GKDD65	99760126K	99	13889K	51	54370049K	3

VOLUME FREE SPACE ALLOC SP LARGEST EXT  
 SERIAL TRK-MANAGD TRK-MANAGD TRK-MANAGED  
 - (2) -- --- (41) --- --- (42) --- --- (43) ---  
 GKDD65 54370270K 13889K 54370049K

**D SMS,OPTIONS**

USEEAV = YES BREAKPOINTVALUE = 21

**ISPF 3.4 output:**

Unit . . : 3390 Free Space

VTOC Data		Total		Tracks		Cyls	
Tracks . . :	180	Size . . . :	1,802,809	120,187			
%Used . . . :	1	Largest . . :	982,545	65,503			
Free DSCBS:	8,996	Free					
		Extents . . :	3				

Volume Data		Track Managed		Tracks		Cyls	
Tracks . . :	1,803,060	Size . . . :	982,549	65,503			
%Used . . . :	0	Largest . . :	982,545	65,503			
Trks/Cyls:	15	Free					
		Extents . . :	2				

Figure 3-2 Empty volume

The LISTCAT output can be confusing, However, remember the 28-bit cylinder number. Figure 3-3 shows the allocated volume after DB2 inserts.

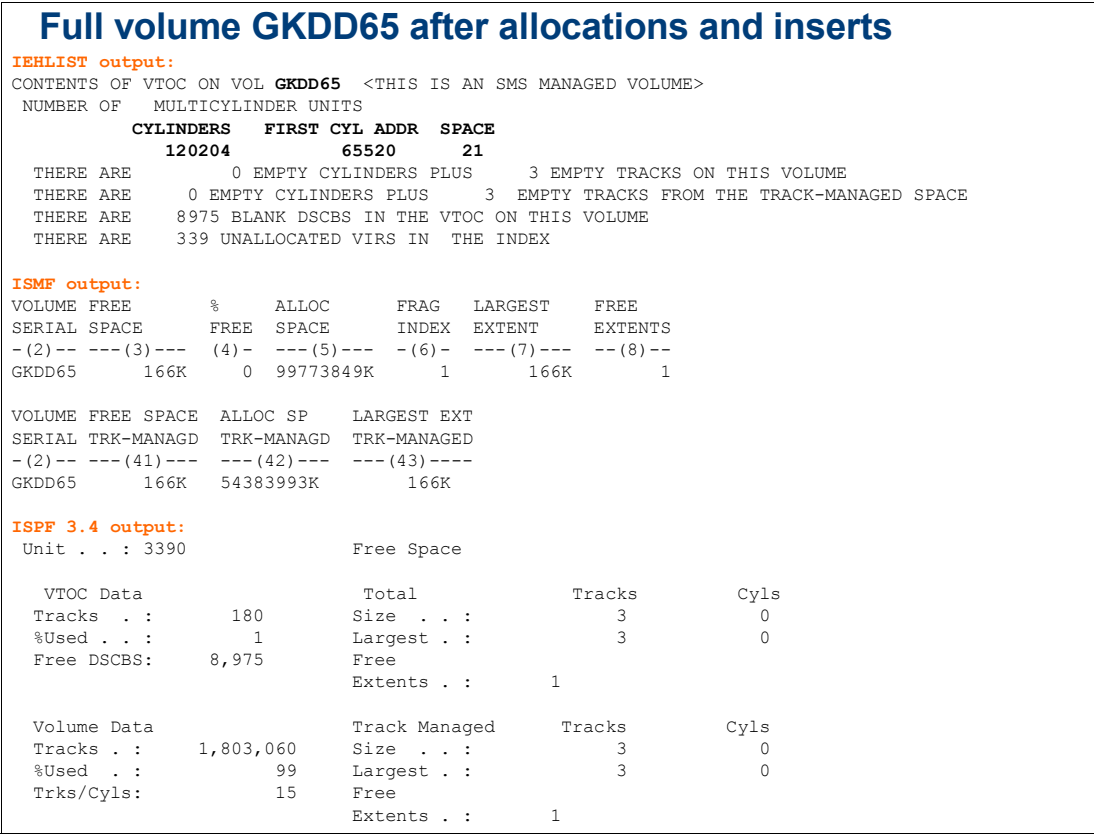


Figure 3-3 Full volume after inserts

Figure 3-4 shows the z/OS Resource Measurement Facility (RMF) format for an EAV volume.

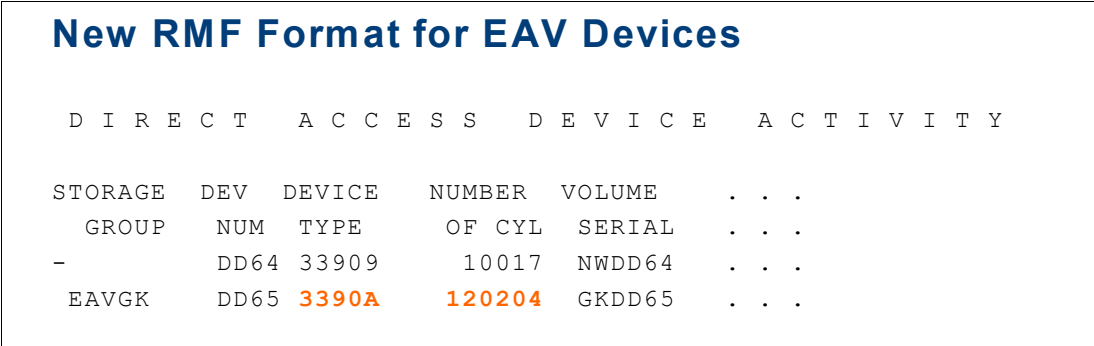


Figure 3-4 New RMF format for EAV devices

### 3.4.4 VSAM striping guidelines for DB2 table spaces

DFSMS striping was first introduced to OS/390® in 1993 and it was successful in two key areas:

- ▶ Avoiding hot spots on individual disks
- ▶ Increasing sequential throughput for individual data streams

Starting in 1993, alternative techniques have been developed to avoid or dissipate a *hot spot*.



RAID 5 is one such technique, helping to spread the I/O for each volume to seven or eight disks.

Storage Pool Striping in the DS8000 control unit is another. Storage Pool Striping spreads a volume across multiple RAID ranks in chunks of 1GB. Storage Pool Striping is a good way of dissipating a hot spot as long as the hot spot is not isolated to a single 1GB chunk.

Therefore, DFSMS striping might not be needed or might not be the best technique for avoiding hot spots.

In 1993, one DB2 dynamic prefetch I/O that read 128 KB required several milliseconds just to transfer the data, although the overhead to start and stop each I/O was comparatively small. The ratio between data transfer time and overhead began to change quickly when FICON was introduced in 2000, and even more quickly as the first decade of the new century began. The initial FICON link speeds were 1 Gbps (gigabit per second), were soon increased to 2 Gbps, and then to 4 Gbps. The z10 processor now supports 8 Gbps channels. Most storage control units do not yet support 4 Gbps speeds (as of 2009), but before long they will. As the link speeds increase relative to the overhead cost of each I/O, minimizing the number of I/Os becomes more important.

However, the I/O quantity (size) of each I/O also plays a role in affecting the performance. For example, although a DS8300 control unit can today read 128 KB in less than one millisecond, reading 256 KB takes more than one millisecond. Now, suppose that we set one millisecond for 128 KB as the *striping objective*. With such criteria, the DS8300 does not need DFSMS striping. However, if our objective is to read 256 KB in one millisecond, two stripes are needed on the DS8300 (so that DFSMS reads 128 KB for each stripe). Alternatively, if our objective is to read 512 KB in less than one millisecond, four stripes are needed.

Now, consider when DB2 uses various data transfer sizes. In all current versions of DB2 (up to and including DB2 9), dynamic prefetch always reads 128 KB, although DB2 utilities always read or write at least 256 KB. In DB2 9, depending on the buffer pool sizes, sequential prefetch can read up to 256 KB and utilities can read and write up to 512 KB. However, if a DB2 query or insert stream or utility is CPU-bound, striping will not make it faster. This situation is common, except when creating or restoring image copies are concerned.

Striping also does not help if any of the components in the data paths are saturated. These paths consist of channels, host adapters, and the bus in each control unit. Take for example the DB2 RECOVER utility. When you want to recover an individual table space, the paths are not saturated and striping reduces the time to restore the table space. However, if you want to recover the entire system, RECOVER utility uses parallelism, and the paths can get saturated, in which case striping might not reduce the time to restore the system.

Another category of I/O that benefits greatly from striping is list prefetch. List prefetch is typical of queries. Log-apply also makes extensive use of list prefetch, as does incremental Copy. Currently, (using 2009 technology), if a table space is striped across a single RAID rank, list prefetch can improve modestly if the devices consist of HDDs, and it can improve significantly if the devices consist of SSDs. Heavy parallelism tends to reduce the striping benefits with HDDs, but striping continues to do well with SSDs under heavy amounts of parallelism.

In conclusion, to summarize the value of striping for table spaces, the most significant benefit occurs with list-prefetch when using solid-state disks, and the applications that benefit the most are long running queries and utilities. Do not overlook the importance of striping table spaces when you need to restore your system from image copies, and to then apply log records, even though your everyday workloads do not necessarily benefit from striping.

As has been true in the past, be aware that hardware technology is forever changing. The value of DFSMS striping may continue to erode.

## 3.5 Large sequential data sets

By default, sequential data sets can allocate up to 4,369 cylinders (65,535 tracks) on one volume. This was a challenge for DB2 when objects grew past this size. Allocation of an image copy data set greater than this size would require the data set be multi-volume.

For example, a mod 9 device with 65,520 cylinders has the capacity to allocate most very large data sets (although it cannot fit a 64-GB data set on one volume; EAV would solve that problem) but would allow the image copy to occupy only 4,369 cylinders of the 65,520 cylinders on each volume.

Data Class allows for an attribute of LARGE for the data set type. The features of the LARGE attribute are as follows:

- ▶ Removes the size limit of 65,535 tracks (4,369 cylinders) per volume for sequential data sets:
  - BSAM, QSAM, and EXCP do not have to be SMS-managed and do not have to be in extended format.  
Data sets do not have to be SMS managed nor in extended format.
  - Architectural limit is 16,777,215 tracks.
  - The limit is still 16 extents per volume. Sequential data sets with EF-enabled allow for 123 extents per volume, which is the same as for VSAM data sets with or without EF enabled. When specifying LARGE as the data set type, you cannot specify EF as the data set type because they are mutually exclusive.
  - When using large sequential data sets, such as with image copies on disk, add DSNTYPE=LARGE on the DD statement.
    - LARGE is not a supported keyword when using TEMPLATE statements for utilities. An alternative is for the TEMPLATE statement to have the DATACLAS keyword that points to a Data Class that has LARGE specified for data set type.
    - The use of MODELDCB does not allow for creating of a large sequential data set. If you have no Data Class assigned with LARGE attribute specified, but you create a model DCB data set with LARGE as the data set type, you cannot code MODELDCB to create the large sequential data set. The data set created in the TEMPLATE statement does not inherit the LARGE attribute of the model DCB data set. The alternative again is to point to a Data Class that has LARGE specified for the data set type.
  - The LARGE attribute can also be specified in the ACS routines using the &DSNTYPE label.
- ▶ In DB2 9, NFM provides support for the following information:
  - Utility input data sets
  - Utility output data sets when coded in the SMS Data Class or DD statement includes DSNTYPE=LARGE

- Archive log data sets, which can be allocated on one disk as multiples of 4 KB up to 4 GB<sup>2</sup>. Archive data sets can be created on disk only in NFM or ENFM modes, however, DB2 tolerates reading them in CM.
- Large sequential data sets are eligible for EAV devices. Do not confuse large sequential data sets with Extended format sequential data sets which are new types of data sets on the EAV introduced by z/OS 1.11.

The next two examples show the image copy of a large table ends in error.

In the first example, Example 3-12, the image copy job did not run, and received a JCL error because of the explicit allocation request for more than 65,535 tracks.

*Example 3-12 COPY utility requesting 7,510 cylinders primary without LARGE specified*

---

```
//UTIL EXEC DSNUPROC,SYSTEM=DB9A,UID='PAULOBROONEY'
//*DSNTICX EXEC PGM=DSNUTILB,PARM='DB9A,IMAGCOPY'
//SYSCOPYX DD DSN=DB9AU.DSNDBD.JOHNIC.TESTLRG5,
//           DISP=(NEW,CATLG,DELETE),
//           SPACE=(CYL,(7510,25))
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
COPY TABLESPACE DSNDB04.JOHNLARG COPYDDN SYSCOPYX
```

```
IEF453I DB2R2B - JOB FAILED - JCL ERROR
IEF344I DB2R2B DSNUPROC UTIL SYSCOPYX - ALLOCATION FAILED DUE TO DATA FACILITY SYSTEM ERROR
IGD17051I ALLOCATION FAILED FOR DATA SET DB9AU.DSNDBD.JOHNIC.TESTLRG5, PRIMARY SPACE EXCEEDS 65535 TRKS
```

---

Next, Example 3-13 differs for the use of TEMPLATE statement. With TEMPLATE, the job does run and later abends with a B37 out-of-space condition for the image copy data set. Because the disposition for the image copy data set was (NEW, CATLG, CATLG), we see that the final allocation was for 4,369 cylinders (65,535 tracks), which is the maximum space allowed for a data set without LARGE attribute specified.

*Example 3-13 COPY utility: TEMPLATE, no space requested, and LARGE not specified*

---

```
//UTIL EXEC DSNUPROC,SYSTEM=DB9A,UID='TEMP'
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
LISTDEF CPY1 INCLUDE TABLESPACES TABLESPACE DSNDB04.JOHNLARG
TEMPLATE LARGE UNIT SYSDA
DSN (DB9AU.DSNDBD.JOHNLARG.TEMPLAT7) DISP (NEW,CATLG,CATLG)
COPY LIST CPY1 COPYDDN (LARGE) SHRLEVEL REFERENCE
```

```
IEC030I
B37-04,IFG0554A,DB2R2B,DSNUPROC,SYS00002,640A,SBOX1X,04210010,DB9AU.DSNDBD.JOHNLARG.TEMPLAT7
IEF450I DB2R2B DSNUPROC UTIL - ABEND=S04E U0000 REASON=00E40347
```

```
DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = TEMP
DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNUGUTC - LISTDEF CPY1 INCLUDE TABLESPACES TABLESPACE DSNDB04.JOHNLARG
DSNUILDR - LISTDEF STATEMENT PROCESSED SUCCESSFULLY
```

---

<sup>2</sup> The ending relative byte address (RBA) of an archive or log data set can be 4GB minus 1 because it starts with 0.

DSNUGUTC - TEMPLATE LARGE UNIT SYSDA DSN(DB9AU.DSNDBD.JOHNLARG.TEMPLAT7) DISP(NEW, CATLG, CATLG)

DSNUJTDR - TEMPLATE STATEMENT PROCESSED SUCCESSFULLY

DSNUGUTC - COPY LIST CPY1 COPYDDN(LARGE) SHRLEVEL REFERENCE

DSNUGULM - PROCESSING LIST ITEM: TABLESPACE DSNDB04.JOHNLARG

DSNUGDYN - DATASET ALLOCATED. TEMPLATE=LARGE

DDNAME=SYS00002

DSN=DB9AU.DSNDBD.JOHNLARG.TEMPLAT7

**DSNUGSAT - UTILITY BATCH MEMORY EXECUTION ABENDED, REASON=X'0B37'**

**DSNUGBAC - UTILITY BATCH MEMORY EXECUTION ABENDED, REASON=X'00E40347'**

ISPF listing for the image copy data set specified in the TEMPLATE statement:

Data Set Name . . . . : DB9AU.DSNDBD.JOHNLARG.TEMPLAT7

General Data	Current Allocation
Management class . . : MCDB22	<b>Allocated cylinders : 4,369</b>
Storage class . . . : DB9A	Allocated extents . : 1
Volume serial . . . : SBOX1X	
Device type . . . . : 3390	
Data class . . . . . : **None**	
Organization . . . : PS	Current Utilization
Record format . . . : FB	<b>Used cylinders . . : 4,369</b>
Record length . . . : 4096	Used extents . . . : 1
Block size . . . . : 24576	
1st extent cylinders: 4369	
Secondary cylinders : 898	Dates
<b>Data set</b> name type :	Creation date . . . : 2010/05/11
SMS Compressible. . : NO	Referenced date . . : 2010/05/11
	Expiration date . . : ***None***

---

Now, we enable the use of large data sets for the sequential output. Example 3-14 allocates the data set adding DSNTYPE=LARGE on the image copy DD statement. We have a successful execution of the image copy job.

*Example 3-14 COPY utility requesting 7,510 cylinders primary with LARGE specified*

---

```
//UTIL EXEC DSNUPROC,SYSTEM=DB9A,UID='PAULOBROONEY'  
//*DSNTICX EXEC PGM=DSNUTILB,PARM='DB9A,IMAGCOPY'  
//SYSCOPYX DD DSN=DB9AU.DSNDBD.JOHNIC.TESTLRGE,DSNTYPE=LARGE,  
//           DISP=(NEW,CATLG,DELETE),  
//           SPACE=(CYL,(7510,25))  
//SYSPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//SYSIN DD *  
COPY TABLESPACE DSNDB04.JOHNLARG COPYDDN SYSCOPYX
```

Allocation was successful and job received a condition code 0.

ISPF of the image copy data set, 4,369 cylinders was exceeded on one volume and the Data set name type is LARGE:

Data Set Name . . . . : DB9AU.DSNDBD.JOHNIC.TESTLRGE

General Data	Current Allocation
Management class . . : MCDB22	<b>Allocated cylinders : 7,510</b>
Storage class . . . : DB9A	Allocated extents . : 1
Volume serial . . . : SBOX1V	

Device type . . . . :	3390	
Data class . . . . :	**None**	
Organization . . . :	PS	Current Utilization
Record format . . . :	FB	<b>Used cylinders . . : 7,250</b>
Record length . . . :	4096	Used extents . . . : 1
Block size . . . . :	24576	
1st extent cylinders:	7510	
Secondary cylinders :	25	Dates
<b>Data set name type : LARGE</b>		Creation date . . . : 2010/05/10
SMS Compressible. . :	NO	Referenced date . . : 2010/05/10
		Expiration date . . : ***None***

---

Example 3-15 points to a Data Class with LARGE enabled by using the DATACLAS keyword on the TEMPLATE statement.

*Example 3-15 COPY utility: TEMPLATE, no space requested, Data Class with LARGE enabled*

---

```
//UTIL EXEC DSNUPROC,SYSTEM=DB9A,UID='PAULOBROONEY'
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
LISTDEF CPY1 INCLUDE TABLESPACES TABLESPACE DSNDB04.JOHNLARG
TEMPLATE LARGE UNIT SYSDA
DSN (DB9AU.DSNDBD.JOHNLARG.TEMPLAT1) DISP (NEW,CATLG,CATLG)
DATACLAS WELCHLRG
COPY LIST CPY1 COPYDDN (LARGE) SHRLEVEL REFERENCE

IGD101I SMS ALLOCATED TO DDNAME (SYS00001)
      DSN (DB9AU.DSNDBD.JOHNLARG.TEMPLAT1          )
      STORCLAS (DB9A) MGMTCLAS (MCDB22) DATACLAS (WELCHLRG)
      VOL SER NOS= SBOX1W
```

Allocation was successful and job received a condition code 0.  
 ISPF of the image copy data set, 4,369 cylinders was exceeded on one volume and  
 the Data set name type is LARGE:  
 Data Set Name . . . . : DB9AU.DSNDBD.JOHNLARG.TEMPLAT1

General Data	Current Allocation
Management class . . : MCDB22	<b>Allocated cylinders : 7,250</b>
Storage class . . . . : DB9A	Allocated extents . . : 2
Volume serial . . . . : SBOX1W	
Device type . . . . . : 3390	
<b>Data class . . . . . : WELCHLRG</b>	
Organization . . . . : PS	Current Utilization
Record format . . . . : FB	Used cylinders . . . : 7,250
Record length . . . . : 4096	Used extents . . . . : 2
Block size . . . . . : 24576	
1st extent cylinders:	7132
Secondary cylinders :	714
<b>Data set name type : LARGE</b>	Dates
SMS Compressible. . :	NO
	Creation date . . . : 2010/05/10
	Referenced date . . : 2010/05/10
	Expiration date . . : ***None***

---

## 3.6 DB2 system table spaces

DB2 uses four databases to control and manage itself and the application data:

- ▶ The catalog database
- ▶ The directory database
- ▶ The work file database
- ▶ The default database

This section provides a general description of these databases.

This section also describes two tables that are directly used by DB2 to manage backup and recovery:

- ▶ SYSIBM.SYSCOPY, belonging to the DB2 catalog
- ▶ SYSIBM.SYSLGRNX, belonging to the DB2 directory

### 3.6.1 The DB2 catalog and directory

The catalog database is named DSNDB06. The directory database is named DSNDB01. Both databases contain DB2 system tables. DB2 system tables store data definitions, security information, data statistics, and recovery information for the DB2 system. The DB2 system tables reside in DB2 system table spaces.

The DB2 system table spaces are allocated when a DB2 system is first created, that is, during the installation process. DB2 provides the IDCAMS statements required to allocate these data sets as VSAM LDSs. The size of these LDSs is calculated from user parameters specified on DB2 installation panels.

Figure 3-5 on page 137 shows panel DSNTIPD with default values for sizing DB2 system table spaces. In this figure, parameters (numbered 1 - 12) are used to size DB2 catalog and directory table spaces.

```

DSNTIPD          MIGRATE DB2 - SIZES
====>
Check numbers and reenter to change:

 1 DATABASES          ===> 200      In this subsystem
 2 TABLES            ===> 20      Per database (average)
 3 COLUMNS            ===> 10      Per table (average)
 4 VIEWS               ===> 3       Per table (average)
 5 TABLE SPACES       ===> 20      Per database (average)
 6 PLANS               ===> 200     In this subsystem
 7 PLAN STATEMENTS     ===> 30      SQL statements per plan (average)
 8 PACKAGES            ===> 300     In this subsystem
 9 PACKAGE STATEMENTS ===> 10      SQL statements per package (average)
10 PACKAGE LISTS       ===> 2       Package lists per plan (average)
11 EXECUTED STMTS      ===> 15      SQL statements executed (average)
12 TABLES IN STMT     ===> 2       Tables per SQL statement (average)
13 USER LOB VALUE STG  ===> 10240    Max KB storage per user for LOB values
14 SYSTEM LOB VAL STG  ===> 2048     Max MB storage per system for LOB values
15 USER XML VALUE STG  ===> 40960    Max KB storage per user for LOB values
16 SYSTEM XML VAL STG  ===> 10240    Max MB storage per system for LOB values
17 MAXIMUM LE TOKENS   ===> 20      Maximum tokens at any time. 0-50

PRESS:  ENTER to continue  RETURN to exit  HELP for more information

```

Figure 3-5 Installation panel DSNTIPD

### 3.6.2 The WORKFILE database

Prior to DB2 9, DB2 supported two databases for temporary files and temporary tables:

- WORKFILE database

This database is used by DB2 for storing created global temporary tables and as storage for work files for processing SQL statements that require temporary working space, for SORTs, materializing views, triggers, nested table expressions, and others.

- TEMP database

This database is used by DB2 for storing the external (user-defined), declared global temporary tables (DGTTS) and the DB2 declared global temporary tables for static scrollable cursor implementation.

The AS clause of the CREATE DATABASE statement with either the WORKFILE or TEMP subclause indicates that the database to be created is either a WORKFILE database or a TEMP database. Each DB2 subsystem or data sharing member has one WORKFILE database and may have one TEMP database.

Starting with DB2 9, the WORKFILE and TEMP databases are converged into one database (the WORKFILE database), and the TEMP database is no longer used. The WORKFILE now also supports SEGSIZE. In DB2 9 conversion mode (CM), the SEGSIZE clause continues to be rejected in the CREATE TABLESPACE statement for creating table spaces in the WORKFILE database. For table spaces that existed in a WORKFILE database before migration from DB2 Version 8, for those created during migration, and for those created in conversion mode of DB2 Version 9, the SEGSIZE column of the catalog table SYSTABLESPACE continues to show 0 (zero) as the segment size. A value of 0 (zero) indicates that these table spaces were created prior to the enablement of the DB2 9

new-function mode. However, DB2 treats these table spaces as segmented, with the default segment size of 16, both in conversion mode (CM) and new-function mode (NFM)

For the work file database, the maximum number of data sets per table space is 32, and the maximum size of a data set is 2 GB in CM and NFM modes

The fields in Figure 3-6 show how to configure the 4 KB and 32 KB page-size table spaces in the DB2 work file database, in the following ways:

- ▶ The total amount of space available for each type of table space
- ▶ The number of table spaces that are to be created during the installation or migration process for each type of table space
- ▶ The segment size of each type of table space (You can change the segment size only in installation mode.)
- ▶ The maximum amount of temporary storage in the work file database that can be used by a single agent at any given time for all temporary tables

In DB2 9, the WORKFILE database is also used by declared global temporary tables and for static scrollable cursor implementation, which used the TEMP database in previous versions. If you created a TEMP database in a previous version of DB2, it is not used in DB2 9. You can drop it anytime after completing migration to DB2 9 conversion mode. If you drop it before entering new-function mode, you must re-create it if you fall back to DB2 V8.

DSNTIP9		INSTALL DB2 - WORK FILE DATABASE	
====>			
Enter work file configuration options below:			
1 TEMP 4K SPACE	====> 20	Amount of 4K-page work space (MB)	
2 TEMP 4K TBL SPACES	====> 1	Number of table spaces for 4K-page data	
3 TEMP 4K SEG SIZE	====> 16	Segment size of 4K-page table spaces	
4 TEMP 32K SPACE	====> 20	Amount of 32K-page work space (MB)	
5 TEMP 32K TBL SPACES	====> 1	Number of table spaces for 32K-page data	
6 TEMP 32K SEG SIZE	====> 16	Segment size of 32K-page table spaces	
7 MAX TEMP STG/AGENT	====> 0	Maximum MB of temp storage space that can be used by a single agent	

Figure 3-6 Install panel for WORK FILE DATABASE

The WORKFILE database supports only table spaces with 4 KB and 32 KB pages sizes:

- ▶ A 4 KB work file table space is used for record size less than 100 bytes.
- ▶ A 32 KB work file table space used for record size greater than or equal to 100 bytes.

When DSNZPARM DSVCI = YES is specified, it enables DB2 to create DB2 managed data sets with a VSAM CI that matches the page size also for the work file.

By having a single converged WORKFILE database, you may define, monitor, and maintain a single WORKFILE database to be used as storage for all temporary files and tables. This convergence preserves the external functionality of the temporary tables as it is today.



**Tip:** The merging of the functions (specified previously) into the single WORKFILE database means the TEMP database is no longer used by DB2. We suggest that you *do not* use DROP on the TEMP database until you are sure that you will not be going back to DB2 V8, otherwise, you have to re-create the TEMP database if you fallback to DB2 V8.

You can add a work file table space or change the size of an existing one by deleting and redefining it. DB2 must be started for these activities and you do not need to stop the WORKFILE database. All DB2 users share the work file database table spaces. You cannot use utilities on the work file database table spaces. You can create additional work file table spaces at any time, including during migration. Creating additional work file table spaces can improve DB2 performance by reducing device contention among applications that require working storage. For information about creating additional temporary work file table spaces, see *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851.

Consider checking or monitoring both the number and size of your current 32 KB work files. We probably have to increase the number of 32 KB work files because of the DB2 9 increase of using 32 KB work files. There is a corresponding decrease in the amount of storage that is needed for 4 KB work files.

Consider also checking the sizes for your 4 KB and 32 KB buffer pools that are used for the work files. We expect that you have to increase the size of your 32 KB buffer pool and decrease the size of the 4 KB buffer pool.

A new online updateable DSNZPARM, MAXTEMPS, has been added to the DSN6SYSP macro. This DSNZPARM parameter specifies the maximum amount of work file storage that an agent can use and is specified in units of megabytes (MB) or gigabytes (GB). If the DSNZPARM value is specified as 0, no limit is enforced, which was the default in previous releases.

When the total storage used by an agent exceeds the MAXTEMPS value that you specified, a Resource Unavailable (DSNT501I) message and Resource Unavailable SQLCODE (-904) error are issued; the activity that caused this condition terminates.

The recent APAR PM02528 introduced a new DSNZPARM DSN6SPRM called WFDBSEP for both DB2 V8 (PTF UK56045) and DB2 9 (PTF UK56046) to provide a hard-separation between DGTT and non-DGTT space used in the work file database. Prior to DB2 9, a natural separation existed between these two groups of functions because DGTTs were contained in the TEMP database (eliminated in DB2 9) and work files were contained in the WORKFILE database.

In DB2 9, users might not be able to limit space usage effectively between DGTTs and work files, such as sort work files, created global temporary tables (CGTT), trigger transition tables, and so on.

## **Monitoring space utilization for the WORKFILE database**

The DSNZPARM MAXTEMPS controls the maximum amount of work file storage an agent can use in megabytes (MB).

The counters for monitoring temporary space utilization at the DB2 subsystem level are included in the IFCID 2 (statistics record). Several counters have been added to the Data Manager Statistics block DSNDQIST to track storage usage in the work file database.

The following fields are written out in IFCID 2:

- ▶ QISTWFCU: Current total storage used, in megabytes
- ▶ QISTWFMU: High watermark, maximum space ever used, in megabytes
- ▶ QISTWFMX: Maximum allowable storage limit (MAXTEMPS) for an agent, in megabytes
- ▶ QISTWFNE: Number of times the maximum allowable storage limit per agent was exceeded
- ▶ QISTWF04: Current total 4 KB page table space storage used, in megabytes
- ▶ QISTWF32: Current total 32 KB page table space storage used, in megabytes
- ▶ QISTW04K: Current total 4 KB page table space storage used, in kilobytes
- ▶ QISTW32K: Current total 32 KB page table space storage used, in kilobytes
- ▶ QISTWFP1: The number of times that a 32 KB page table space was used when a 4 KB page table space was preferable (but not available)
- ▶ QISTWFP2: The number of times that a 4 KB page table space was used when a 32 KB page table space was preferable (but not available)

### DB2 installation panels: SMS related

In the panels described in this section, we indicate the options that are introduced in DB2 9 and that affect the DB2 ICF Catalog and SMS-related options. Figure 3-7 on page 141 shows the panel where you describe the VSAM data set cataloging options.

The installation jobs classify the DB2 VSAM data sets, which include recovery log, subsystem, and user data sets. You must create the catalog that defines these data sets through the VSAM ICF.

**Tip:** Consider using an ICF user catalog to classify all DB2 objects because you can use aliases for user catalogs. When you use the CREATE STOGROUP statement, you might need to use an alias for the VCAT option, which must be a single-level one to eight character name. You can use a master catalog, but only if the name of the master catalog is a single-level name of one to eight characters

DB2 does not require you to put all DB2 VSAM data sets in the same ICF catalog - you should strongly consider *not* to have all your VSAM data sets in the same ICF catalog. In this example, the catalog that you create when installing is called the primary ICF catalog. You must put some data sets in the primary catalog. You can put other data sets elsewhere; some data sets do not have to be classified at all.

The entries on the panel of Figure 3-7 on page 141 allow you to define the alias of the VSAM catalog for DB2 subsystem data sets, whether you want DB2 to define a new ICF catalog, and whether data sets and DB2 storage groups that are created by DB2 installation, migration, and verification (IVP) jobs are to be defined on disk storage that is controlled by the z/OS Storage Management Subsystem (SMS).

<b>DSNTIPA2</b>	INSTALL DB2 - DATA PARAMETERS PANEL 1	
====>		
Check parameters and reenter to change:		
1 CATALOG ALIAS	====>	DB29A Alias of VSAM catalog for DB2 subsystem data sets
2 DEFINE CATALOG	====>	YES YES or NO
3 USE SMS	====>	YES For installation-defined objects: NO, YES

Figure 3-7 Data parameters for DB2 installation panel 1

Option 3 USE SMS denotes whether the data sets and DB2 storage groups that will be created during installation and migration are to be controlled by SMS. This step includes the user-managed data sets for the VSAM catalog, DB2 catalog, directory, default database, LOG files and the BSDS.

A value of NO indicates that the data sets and storage groups are to be allocated on specific volume serials that you identify on the next panel, shown in Figure 3-8 on page 142.

A value of YES indicates that the data sets and storage groups are to be allocated on SMS-managed storage, using any SMS data classes, management classes, and storage classes that you optionally specify on the next pane. See Figure 3-8 on page 142.

**Note:** Do not enter YES in the USE SMS field unless all of the following statements are true:

- ▶ SMS is active on the system where this instance of DB2 resides or is being installed.
- ▶ Your system storage administrator has provided the prerequisite SMS automatic class selection (ACS) routines for data sets and DB2 storage groups that are created during DB2 installation or migration.
- ▶ Your system storage administrator has provided any SMS data classes, management classes, or storage classes that you plan to specify on panel DSNTIPA3

The BSDS is VSAM KSDS. The archive logs are sequential. All other data sets are VSAM linear data sets.

Figure 3-8 shows how the size and number of table spaces in the work file database are defined.

```

DSNTIPA3                                INSTALL DB2 - DATA PARAMETERS PANEL 2
====>

Check parameters and reenter to change:
 1 PERMANENT UNIT NAME    ==> 3390                Device type for MVS catalog
                                                    and partitioned data sets
 2 TEMPORARY UNIT NAME    ==> SYSDA                Device type for
                                                    temporary data sets

                                                    ----- SMS -----
                                                    DATA CLASS  MGMT CLASS  STOR CLASS
                                                    -----
 3 CLIST ALLOCATION        ==> VOLUME1 ==>          ==>          ==>
 4 NON-VSAM DATA         ==> VOLUME2 ==>          ==>          ==>
 5 VSAM CATALOG, DEFAULT, ==> VOLUME3 ==>          ==>          ==>
   AND WORK FILE DATABASE
 6 DIRECTORY AND CATALOG ==>          ==>          ==>          ==>
   DATA
 7 DIRECTORY AND CATALOG ==>          ==>          ==>          ==>
   INDEXES
 8 LOG COPY 1, BSDS 2     ==>          ==>          ==>          ==>
 9 LOG COPY 2, BSDS 1     ==>          ==>          ==>          ==>

```

Figure 3-8 Installation panel DSNTIPA3 for defining subsystem databases and data sets

Table 3-4 lists the acceptable values for the non-VSAM, VSAM and LOG data sets.

Table 3-4 Acceptable values for the data values in Figure 3-8

	VOLSER	DATA CLASS	MGMT CLASS	STOR CLASS
Valid values if using SMS	Blank or VOLSER #	Blank or Data class name	Blank or valid SMS management class	Blank or valid SMS Storage class name
Valid values if <i>not</i> using SMS	Valid VOLSER #	Blank	Blank	Blank
Default if using SMS	Blank	Blank	Blank	Blank
Default if <i>not</i> using SMS	DSNV01 or DSNV02	Blank	Blank	Blank

### 3.6.3 SYSIBM.SYSCOPY

The SYSIBM.SYSCOPY table is a DB2 catalog table. It is referred to by its short name SYSCOPY. The table space in which SYSCOPY is stored is DSNDDB06.SYSCOPY.

SYSCOPY table contains recovery-related information for each table space. Although its main purpose is to manage image copies, other related recovery information is also recorded here.

For example, SYSCOPY contains information of the following items:

- ▶ Image copies
- ▶ Quiesce points
- ▶ LOAD executions
- ▶ REORG executions
- ▶ RECOVER executions

Table 3-5 lists the values for the image copy type information for COLUMN ICTYPE.

Table 3-5 *SYSIBM.SYSCOPY: ICTYPE column values*

ICTYPE	Description
A	An ALTER was performed.
B	A REBUILD INDEX was performed.
C	A CREATE was performed.
D	CHECK DATA LOG(NO); No log records for the range are available for RECOVER utility
E	RECOVER (to current point)
F	COPY FULL YES
I	COPY FULL NO (Incremental)
M	MOFIDY RECOVER utility
P	RECOVER TOCOPY or RECOVER TORBA (partial recovery point)
Q	QUIESCE
R	LOAD REPLACE LOG(YES)
S	LOAD REPLACE LOG(NO)
T	TERM UTILITY command performed
V	REPAIR VERSIONS utility executed
W	REORG LOG(NO)
X	REORG LOG(YES)
Y	LOAD LOG(NO)
Z	LOAD LOG(YES)

In addition to the SYSCOPY column ICTYPE (which is used to record recovery-related events) the column STYPE is used in conjunction for certain recovery scenarios.

The SYSCOPY column, PIT\_RBA, is used to store the finish time of a SHARE LEVEL CHANGE copy. The Recover utility uses this value to locate the correct image copy to start recovery when recovering to certain timestamp.

The SYSCOPY column, STYPE, is used to remember whether using the RECOVER utility to recover to a point in time (PIT) was done with or without consistency:

- ▶ If ICTYPE is P, and STYPE is blank, recovery to PIT was done *without* consistency
- ▶ If ICTYPE is P and STYPE is C, recovery to PIT was done *with* consistency

### 3.6.4 SYSIBM.SYSLGRNX

The SYSIBM.SYSLGRNX table is a DB2 directory table. It is referred to by its short name, SYSLGRNX. The table space in which SYSLGRNX is stored is DSNDB01.SYSLGRNX.

The purpose of SYSLGRNX is to identify, on the DB2 log, where the updates are for each object.

**Note:** Seven data sets do not have SYSLGRNX entries: five table spaces of the DB2 directory (DSNDB01), DSNDB06.SYSCOPY, and DSNDB06.SYSGROUP.

These are referred to as *always open*. To recover any of them, we must pass all of the log since the image copy was taken, whether or not any updates ever occurred.

The SYSLGRNX table stores records that serve to improve recovery performance by limiting the scan of the log to changes that must be applied. The SYSLGRNX records contain the first log RBA (or LRSN in a data sharing group) and the last log RBA (or LRSN) of updates to a table space. The record is opened when a first-update is detected, and closed after an interval of read-only activity. The interval is defined with two read-only switch parameters, set by the fields RO SWITCH CHKPTS and RO SWITCH TIME on the DB2 installation panel DSNTIPN.

Observe that each entry in the SYSLGRNX directory table reflects updates by a specific member in data sharing. This means that in a four-way data sharing group, each of the four can have entries for the same times. Entries can explode in SYSLGRNX extents if a user is not diligent about pruning old entries. This activity is accomplished using the MODIFY utility, which deletes old image copies and the SYSLGRNX entries that apply to them

Use the MODIFY utility to maintain or manage the information in SYSLGRNX. Introduced in DB2 9, the MODIFY utility has been updated to perform the following tasks:

- ▶ Delete SYSLGRNX records according to AGE or DATE criteria even if there are no SYSCOPY records to delete.
- ▶ Inserts a new SYSCOPY record ICTYPE=M, STYPE=R, with START\_RBA equal to the highest RBA/LRSN of the SYSCOPY or SYSLGRNX records that are deleted.
- ▶ Deletes SYSCOPY records based on the following information:

<b>RETAIN LAST <i>n</i></b>	Keep last <i>n</i> image copies.
<b>RETAIN LOGLIMIT</b>	Based on archive logs in the BSDS.
<b>RETAIN GDGLIMIT</b>	Based on the image copy GDG limit.

Stopping SYSLGRNX entries are recorded when you use STOP the data set, STOP DB2, and QUIESCE WRITE(YES) the data set. It also occurs during Pseudo Close. If DB2 terminates abnormally, none of the data sets open for a given DB2 (member) are closed. This condition leaves SYSLGRNX with a valid starting entry, but adds zero to the closing entry. If this condition is not resolved, recovery cannot complete successfully.

In this case, DB2 applies the log from the starting RBA/LRSN to *currency* or to the next starting/stopping pair. This step encompasses minutes or many hours of log data that must be passed. So, during end-restart, the phase that follows backward recovery (Phase 4), DB2 updates all the data sets, which have stopping entries that are still zeroes, and thus closes out the start/stop pair. During end-restart, DB2 performs other housekeeping tasks necessary to be ready to use, when restart is complete.

**Important:** If you rely on FlashCopy or other forms of *instant* copies as your back up and never run the COPY utility, your SYSLGRNX entries cannot be pruned. Only an image copy recorded in SYSCOPY allows MODIFY utility to be successful, so at least one valid copy must exist. Many users are not aware of this situation until the ever-increasing extents of SYSLGRNX are exposed. The MODIFY utility discards SYSLGRNX records that meet the deletion criteria when the AGE or DATE options are specified, even if no SYSCOPY records were deleted.

## 3.7 DB2 application table spaces

All application data in DB2 is organized in the objects described in 3.3, “DB2 data objects” on page 117. Application table spaces and index spaces are created as described in 3.4, “Creating table spaces and index spaces” on page 127.

Application table spaces and index spaces are also VSAM LDS data sets, with exactly the same data attributes as DB2 system table spaces and index spaces. The distinction between system and application data is made only because their performance and availability requirements differ.

The partitioned page set assumes the following name:

**catname.DSNDBx.dbname.psname.y000z.1nnn**

The name has the following definitions:

<b>catname</b>	The high-level qualifier (HLQ).										
<b>x</b>	Can be C (for VSAM cluster) or D (for VSAM data components).										
<b>dbname</b>	The DB2 database name.										
<b>psname</b>	The table or index space name.										
<b>y000z</b>	Can be I or J, which indicates the data set name used by REORG with FASTSWITCH.  Starting with DB2 9, the number ( <b>z</b> ) can be 1 or 2, making the qualifier either I0001, I0002, J0001, or J0002.										
<b>1nnn</b>	The <b>1</b> can be A, B, C, D, or E, and the <b>nnn</b> is the data set number beginning with 001. Non-partition data sets always start with A001, for partition data sets:  <table><tr><td>A001-A999</td><td>for partitions 1 - 999</td></tr><tr><td>B000-B999</td><td>for partitions 1000 - 1999</td></tr><tr><td>C000-C999</td><td>for partitions 2000 - 2999</td></tr><tr><td>D000-D999</td><td>for partitions 3000 - 3999</td></tr><tr><td>E000-E096</td><td>for partitions 4000 - 4096</td></tr></table>	A001-A999	for partitions 1 - 999	B000-B999	for partitions 1000 - 1999	C000-C999	for partitions 2000 - 2999	D000-D999	for partitions 3000 - 3999	E000-E096	for partitions 4000 - 4096
A001-A999	for partitions 1 - 999										
B000-B999	for partitions 1000 - 1999										
C000-C999	for partitions 2000 - 2999										
D000-D999	for partitions 3000 - 3999										
E000-E096	for partitions 4000 - 4096										

## 3.8 Index compression

In data warehouse type applications, it is common to have many large indexes defined on large tables. It is also possible to have index spaces that are much larger than the tables they are based upon. To lower the cost of ownership and to improve scalability (more index entries in a single physical index data set), in DB2 9 NFM, you may compress your indexes and you may define indexes with page sizes larger than 4 KB.

Unlike table compression, the compression mechanisms implemented for index compression do not require a compression dictionary. Because DB2 performs dictionary-less compression, newly created indexes may begin compressing their contents immediately. Implementation of index compression only compresses the data in the leaf pages. The goal is to reduce the cost of ownership, therefore index pages are stored on disk in their compressed format (physical 4 KB index page on disk) and are expanded when read from disk into 8 KB, 16 KB, or 32 KB pages. To turn on compression for an index, it must be defined in an 8 KB, 16 KB, or 32 KB buffer pool.

The CREATE INDEX and ALTER INDEX now have a new keyword COMPRESS YES/NO. If you ALTER INDEX COMPRESS YES/NO, the index is placed in REBUILD-pending status.

Compression takes place for the whole index. It cannot be activated at the partition level.

As with table compression, you can use DSN1COMP to estimate the effectiveness of compressing an index. This approach helps you decide which indexes can benefit from compression. No parameters are required and only one parameter is allowed for index estimates, PARM='LEAFLIM(x)', where x is the number of leaf pages to be scanned. If you want to scan all index pages, do not specify the LEAFLIM parameter. The estimate will provide you with an evaluation of compression using various index page sizes. It is important that you do not choose a larger index page size than what is recommended. If you do, the result can be the wasting of valuable buffer pool space. As the page size increases, more unused space exists. In this example, specifying an 8 KB page size might be the best solution, because specifying a larger page size can waste too much space. A guideline is to stop below 50% unused space when choosing the compressed index page size.

**Note:** COPY of a compressed index creates an uncompressed format output file. This likely causes the size of the space required to store the image copy data set to exceed the size of the source index

Data compression uses the standard Ziv-Lempel compression algorithm, which requires a dictionary. DB2 uses a Ziv-Lempel hardware instruction called CMPSC that is built into the z architecture, which provides DB2 for z/OS with excellent compression performance. Details about CMPSC can be found in the *z/Architecture Principles of Operations*, SA22-7832. Because the buffer pool contains compressed data, every time that a row is fetched from a page, the row has to be decompressed before being passed to the application. Because buffer pools contain compressed data, DB2 can do I/O directly in and out of a buffer.

The CPU cost of data compression is not significant for typical online transaction processing (OLTP) work, which randomly fetches one row for each Getpage. However, the cost of data compression for sequential applications (for example, table scans, sequential inserts and utilities) can be significant.

### Data compression requires a dictionary

Because data compression requires a dictionary, inserted rows cannot be compressed until the dictionary is built for a table (or table partition). Only the utilities Load and Reorg are capable of building the compression dictionary, and there is some amount of overhead to do this. To minimize this overhead, the utilities provide the KEEPDICTIONARY keyword, if reusing an old dictionary is acceptable. However, if the nature of the data has changed, reusing the old dictionary can result in sub-optimal compression.

For more information about data compression, see *DB2 for OS/390 and Data Compression*, SG24-5261.



### Index compression is without a dictionary

Many differences exist between data compression and index compression. A major difference is that index compression does not use a dictionary. One advantage of not having a dictionary is that DB2 can compress newly inserted index keys immediately without waiting for LOAD or REORG to be run. In fact, many applications use SQL to load a database rather than LOAD. Because index compression does not use a dictionary, the REORG INDEX utility never has to be run for building a dictionary.

## 3.9 DB2 recovery data sets

To ensure data integrity, DB2 uses several traditional data sets for recovery purposes. Not all of these are always needed by DB2, but all are required for contingency reasons. DB2 supports two or more copies of these data sets to ensure a high level of data integrity.

A short description of DB2 recovery data sets is provided here. A more detailed description is available in “Managing the Log and the Bootstrap Data Set” section of *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840.

### 3.9.1 Bootstrap data sets

DB2 uses the bootstrap data set (BSDS) to manage recovery and other DB2 subsystem-wide information. The BSDS contains information needed to restart and to recover DB2 from any abnormal circumstance. For example, all log data sets (active and archive) are automatically recorded within the BSDS. While DB2 is active, the BSDS is open and is updated.

Although you may define a single BSDS, but because the BSDS is critical for DB2 data integrity, the presence of two copies of the BSDS at start time is a better practice. If a copy fails while DB2 is running, DB2 sends a warning message and continues operating with a single BSDS. It is the responsibility of operations to monitor this circumstance and restore the BSDS duality as soon as possible.

To recover a lost BSDS, when DB2 *is executing*, perform the following steps:

1. Delete the failing BSDS.
2. Redefine the failing BSDS, or alternatively, rename an existing spare BSDS.
3. Rebuild the BSDS by using a **-RECOVER BSDS** command.

If a BSDS copy fails while DB2 is starting, the startup does not complete.

To recover a lost BSDS, when DB2 *is stopped*, perform the following steps:

1. Delete the failing BSDS.
2. Redefine the failing BSDS, or alternatively, rename an existing spare BSDS.
3. Rebuild the BSDS from the good copy with an IDCAMS REPRO command.

In DB2 9, the BSDS must be in the format that supports up to 10,000 data sets per copy for archive logs and 93 per copy for active logs. The BSDS is converted with an optional conversion job supplied in the DSNTIJUZ procedure when the install CLIST is run in MIGRATE mode. In versions before DB2 9, when a new BSDS is created and formatted using DSNJU003, it is in the old format that allows only 31 active logs and 1,000 archive logs. If a site wants to use the new format for a new BSDS, it must create the BSDS, format it using DSNJU003, and then run the DSNJCNVB program to convert it to the new format.

When migrating from DB2 V8 to DB2 9, running the conversion program DSNJCNVB is required if not previously done when running DB2 V8 in new-function mode. If the conversion of the BSDS is not performed prior to migration and if the BSDS is in the old format, a new message, DSNJ1571, is issued during DB2 startup, and startup is terminated. Also, MAXARCH DSNZPARM now defaults to 10,000 for new installs.

**Important:** There are no fallback or coexistence considerations because DB2 V8 NFM has always supported the new format. Therefore, be sure to make a backup copy of your BSDS before running the conversion utility.

### 3.9.2 Active logs

The active log data sets are used for data recovery and to ensure data integrity in case of software or hardware errors. DB2 uses active log data sets to record all updates to user and system data.

The active log data sets are open as long as DB2 is active. Active log data sets are reused when the total active log space is used up, but only after the active log (to be overlaid) has been copied to an archive log.

DB2 supports dual active logs. Be sure to make use of dual active logs for all DB2 production environments.

#### Sizing active logs

DB2 logs always use 4 KB pages. For extremely massive sequential inserts, logging might become an I/O bottleneck with large row sizes and infrequent commits. Small row sizes might stress first the log latch and the CPU. Measurements were done with 4000 byte rows sequential inserts generating large amount of log data with minimal CPU time.

Prior to MIDAWs, the maximum log throughput using the DS8000 and FICON Express 2 was 84 MBps, and striping the log increased the bandwidth only slightly. MIDAWs increased the log bandwidth by 31%, reaching 116 MBps with two stripes.

The amount of space dedicated to each individual active log data set is not critical for the DB2 administrator. Traditionally, the active logs have been sized for practical reasons, for example, to make best use of the archive log device (tape cartridge or disk volume).

The overall size of all active log data sets is important for the DB2 DBA. This size plays a critical role in the backup and recovery strategy.

The number of active log data sets, multiplied by the space of each active log, defines an amount of log information most readily available: the capacity of the active log. This capacity defines the time period that has the best recovery performance and the highest data availability service level. The reason is that the DB2 RECOVER utility generally performs better with an active log than with an archive log. See 7.5.2, “Active log size” on page 363 for more details.

#### Effect of log size on backup and recovery strategy

The relationship between the types of log data sets is shown in Figure 3-9 on page 149. This figure shows a timeline that begins when a DB2 subsystem is first started (Start Time) and proceeds until the current time (Current Time). During this whole time, log data has been generated; this is shown by the DB2 LOG bar.

The log data sets have limited capacity and cannot cover the total time period.

The amount of DB2 log in the active log data sets (the active log capacity) is shown as the time period from Time 2 to the Current Time. The oldest, still available archive log corresponds to Time 1. Because the whole log is not available, recoveries are only possible throughout the period from Time 1 to Current Time. The time period from Time 2 to Current Time corresponds to the period with most efficient recoveries because, generally, the active log is allocated on faster devices. The archive log usually overlaps with the active log for a minimum of the last pair of active log data sets not yet archived up to some time after Time 2 and before Current Time. If the data needed for RECOVER or RESTART has been archived, but is still available on an active log data set not yet reused, DB2 accesses the active log.

A good backup and recovery strategy considers the following information:

- ▶ The amount of time to cover with all logs (Time 1 up to Current Time)
- ▶ The amount of time to cover with active logs (Time 2 up to Current Time)

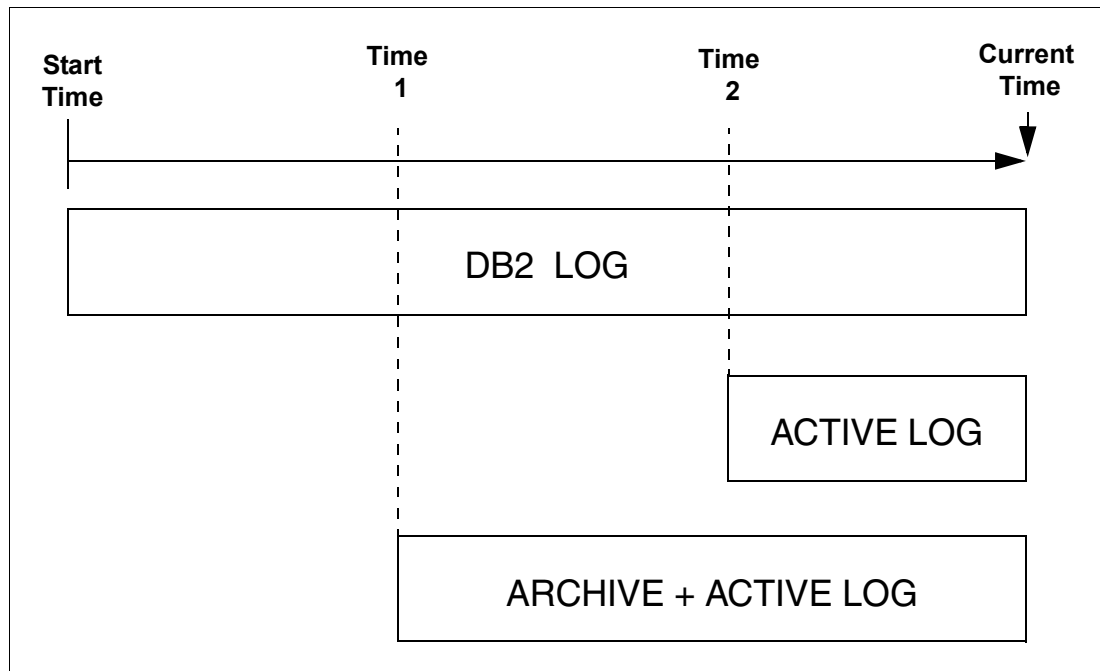


Figure 3-9 DB2 log and its data sets

## Active log performance

IBM made big breakthroughs in sequential I/O performance when it introduced Adaptive Multi-Stream Prefetching (AMP), available with the 2.4G Licensed Internal Code release of the DS8000 (DS8000 R2.4.2 Maintenance Release Letter: R11p.9b070621 (Bundle 62.42.77.0)).

AMP optimizes cache efficiency by incorporating an autonomic, workload-responsive, self-optimizing prestaging technology. The new algorithm dynamically decides when and what to prestage, delivering up to a two-fold increase in the sequential read capacity of RAID 5 arrays. AMP improves the prestaging performance of the DS8000, thereby enabling a DB2 sequential operation to more fully take advantage of FICON Express 4 channels and the fast host adapters in the DS8000 Turbo.

Using DB2 V8, prior to AMP, the DS8000 was able to prestage 4 KB pages at 92 MBps. The DS8000 host adapter was able to transfer this data across the FICON link at about 97 MBps, and the DS8000 Turbo host adapter was able to transfer this data at 132 MBps. Nevertheless, prior to AMP, the prestaging rates of the DS8000 Turbo for DB2 V8 were limited to 92 MBps.

AMP increases the prestaging rate, enabling DB2 Version 8 faster sequential access, up to 130 MBps.

Striping a data set across multiple RAID 5 disk arrays using DFSMS enables yet higher data transfer and prestaging speeds. How to activate striping is discussed at 4.3.3, “SMS storage class” on page 222.

### DB2 log striping guidelines

During normal OLTP operations, DB2 active logs are written and are usually read only to archive the log records. Most log writes are asynchronous I/Os with respect to the application and each I/O consists of a small number of 4 KB pages. However the log I/O must have completed at COMMIT to allow for data integrity. Much has been said about how faster channels increase log bandwidth; a DS8300 control unit can achieve 116 MBps, without striping, and the maximum bandwidth with striping is a little bit higher. However, most OLTP workloads never push the log throughput beyond 10 MBps. Even if the log latch is not a bottleneck, the I/O response times for online transactions gradually degrade when log buffers queue up waiting for the previous I/O to complete. Data sharing may be used to overcome log latch contention or queuing of log buffers. Also, DB2 9 helps to relieve log latch contention for an individual DB2 member by allowing the assignment of log record sequence numbers unique within a data or index page rather than the subsystem.

As technology has evolved during the last several years, corporations have begun to use remote replication to protect themselves against disasters. Remote replication is often the cause of I/O performance problems. Not all storage control units perform remote replication the same way. The performance of remote replication is such a critical factor in I/O performance that it should be one of the most important factors influencing the purchase choice of a storage control unit.

Because all log writes caused by OLTP are synchronous, log I/O is susceptible to problems caused by remote replication. Each I/O introduces a risk of another delay, and because striping causes more I/Os, striping increases the risk of a delay. Remote replication is not the only thing that causes stress on a control unit, but it is one of the biggest causes.

Striping the active log would help the performance of active log reads with DB2 9, because DB2 9 increased the I/O quantity for active log reads from 15 pages to 120 pages. However, given 4 Gbps links, and future hardware that will use 8 Gbps links, the risks of striping the active logs likely outweigh the benefits.

### 3.9.3 Archive logs

Archive log data sets, managed by DB2, are backups of the active log data sets. Archive log data sets are created automatically by DB2 when an active log is filled. They may also be created with the **-ARCHIVE LOG** command for operational requirements. Additional circumstances can trigger the archiving process. The *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840, describes these in detail.

DB2 supports dual archive logs; use dual archive log data sets for all production environments. When dual archive log is specified, during the archiving, the primary active log is read and two archive log data sets are written in parallel. For better archive-log availability, define both copies on separate devices (or SMS classes) to physically separate the dual data sets.

Archive log data sets are required for any recovery that spans a period of time in excess of the time covered by the active logs. *Archive log data sets* are sequential data sets that have the following features:

- ▶ They can be defined on disk (good practice) or on tape.
- ▶ They can be migrated.
- ▶ They can be deleted with standard procedures.

Archive log data sets are required for data integrity. Procedures are required to ensure that archive log data sets are only deleted when they are not going to be required anymore.

Information related to archive logs are in the following sections:

- ▶ “Effect of log size on backup and recovery strategy” on page 148
- ▶ “Deleting image copies and archive logs” on page 160

DB2 9 for z/OS implemented various logging enhancements to remove previous restrictions that caused bottlenecks in logging activity. In this section, we discuss these enhancements.

### Remove data sharing LRSN increment wait

In data sharing prior to DB2 9 for z/OS, the log record sequence number (LRSN) value of each log record on a given member had to be unique. If two successive log records for the same member had the same LRSN value, DB2 would redrive the store clock (STCK) for the LRSN until a different one was generated. The LRSN value increments every 16 microseconds. On today's processors, 16 microseconds is a long time to wait, and can cause significant delays for workloads that have a high logging rate. In DB2 9 for z/OS NFM and ENFM modes, LRSN values have to be unique only within a given page. That is, successive updates, inserts, and deletes into a given page must have unique LRSNs to guarantee the order the changes arrived.

So, DB2 9 for z/OS (NFM and ENFM) drives the LRSN increment wait only when successive updates, inserts, and deletes into a given page have the same LRSN value. With this enhancement, DB2 also removes the need to keep holding the log latch (LC19) while the LRSN is incremented. Log latch contention may be significantly reduced or eliminated for data sharing workloads that were experiencing contention for all cases but the intensive multi-row updates.

**Note:** DSN1LOGP output may now include multiple log records with the same LRSN value on a single DB2 data sharing member. DSNJU004 might list consecutive active or archive log data sets where the end LRSN value of the first is the same as the beginning LRSN value of the second. These conditions can occur in DB2 9 for z/OS NFM, ENFM, or in CM after falling back to that mode.

### Convert disk archive log reads from BDAM to BSAM

Prior to DB2 9 for z/OS, all disk<sup>3</sup> archive log reads were done using basic direct access method (BDAM). Although BDAM allows for rapid access to the target CI, it does not provide support for striped data sets or DFSMS DASD compression (Extended Format data sets). Because active logs support Extended Format data sets and can be striped, it takes longer to write an archive log copy than the active log that the archive data is coming from. If this situation continues on a busy system for a sufficient period of time, the possibility exists that logging can be suspended while waiting for the archive log processing to catch up.

<sup>3</sup> Tape archive log reads have always used BSAM.

Beginning with conversion mode (CM), DB2 9 for z/OS uses basic sequential access method (BSAM) to read DASD archive logs (QSAM is still used to write archive logs). In DB2 9 for z/OS NFM these data sets can be defined as Extended Format data sets. This allows DASD archive logs to be striped and compressed. With striped archive logs, DB2 can now match the performance of striped active logs, thus ensuring that as logging volumes increase, log archiving can scale at the same rate.

**Note:** After converting to NFM, if you decide to have archive logs striped or compressed, any recovery actions involving those archive logs must be performed on a DB2 9 system. You cannot create archive logs as extended format data sets while in CM, but reading them in CM is tolerated. Extended Format data sets must be SMS managed.

### **Convert archive log processing from AMODE(24) to AMODE(31)**

Prior to DB2 9 for z/OS, certain customers had reported running short of storage below the 16 MB line in the MSTR address space. Part of the problem was that DB2 used 24-bit mode to process log buffers that were used to read and write archive logs, and this process required to define them below the 16 MB line (and also significantly limited the size of these buffers).

To alleviate the storage constraint and to provide for increased archive log processing, archive log processing (reads and writes) in the MSTR address space is converted from 24-bit mode to 31-bit mode.

By converting to using 31-bit mode, DB2 can now move these buffers above the 16 MB line, allowing them to be much larger. In addition, DB2 now uses dual buffering (fill the next buffer while processing the current one) for archive log reads when possible and the size of the log buffers has been greatly increased. This situation is expected to result in improvements of archive log read and write performance and throughput. This enhancement is in effect starting with DB2 9 in conversion mode.

### **Use z/OS DSNTYPE=LARGE disk data set support**

Since the inception of the xSAM access methods, there has been a sequential and partitioned data set size limit of 64 KB tracks per DASD volume. z/OS 1.7 lifts this limit with the introduction of DSNTYPE=LARGE data set support. The new limit for these types of data sets is 16 million tracks on a single DASD volume (or across 59 DASD volumes). DB2 currently supports active logs as large as 4 GB. You can define them larger but DB2 only uses 4 GB.

For 3390 geometry DASD, the requirement is 87,382 tracks, and for 3380 geometry the requirement is 104,858 tracks. Both of these are above the 64 KB track limit, forcing installations that use 4 GB logs to put archive logs on tape or to use multi-volume archive logs. By using the new z/OS support for large data sets, installations will be allowed to have 4 GB DASD archive logs. This may simplify management of DASD archive logs because 4 GB active logs will no longer require multi-volume allocations. You can create these large DASD archive logs only in NFM or ENFM modes. However, DB2 tolerates reading them in CM.

## **3.9.4 Image copies**

Image copies are the backup of user and system data. Image copies of index spaces are also now being considered for use for large spaces. For a well-managed backup and recovery policy, be sure that the amount of data in image copy data sets covers at least three generations of image copies to guarantee recoverability. This means that a large number of image copy data sets is required and must be managed in DB2 installations.

## Image copy availability

Image copies ensure user and system data integrity. Their availability is critical for DB2 system and application availability. DB2 can optionally generate up to four image copies of a table space, index space, or data set (for a multiple data set table space or index space). Two of these copies are intended for a disaster recovery at a remote site. For better image copy availability, customers should define the copies on different devices (or SMS classes) to physically separate the data sets.

## Image copy options

Image copies can be run in either of two important varieties:

- *Full image copies* are complete backups of a table space or data set.
- *Incremental copies* only contain the changes since the last full image copy.

Incremental and image copies can be combined (merged) to create other incremental or full image copies.

The SHRLEVEL option is used to specify application access during the copy. SHRLEVEL REFERENCE creates a consistent copy. During the SHRLEVEL REFERENCE copy, only read access is allowed. SHRLEVEL CHANGE creates a copy while the data is updated.

We examine improvements related to the COPY utility. Note that the COPY utility now uses the most recently used (MRU) algorithm for the pages that it reads in, thus preventing trashing of the buffer pool.

## COPY with CHECKPAGE

The CHECKPAGE option is generally recommended in best practices. Prior to DB2 9, COPY had two disadvantages. The first one is the relatively expensive CPU overhead (about 5% for COPY INDEX and 14% for COPY TABLESPACE with DB2 V8). The second one is that, if COPY finds a broken page, it puts the table space or index space into COPY PENDING state and making it inaccessible to further user operations while it continues to look for broken pages until the end of the object.

With DB2 9, work has been done to reduce the CPU overhead for COPY TABLESPACE with CHECKPAGE to be almost negligible. Furthermore, the buffer steal algorithm used by the COPY utility has been changed from LRU to MRU buffer. This change both reduces CPU time and prevents the content of the buffer pool from being dominated by the image copy.

With DB2 9, the CHECKPAGE option is always in operation. The COPY utility performs validity checking for each page, one page at a time, in the table space or index space. If it finds an error, COPY issues message DSNU518I, which identifies the broken page and the type of error. If more than one error exists in a page, only the first error is identified. COPY continues checking the remaining pages, but does not copy them, in the table space or index space after it finds an error.

When the error has been detected, the object table space or index space is not put in COPY pending state. Instead, a return code of 8 is issued. When COPY has detected a broken page, and all the following pages have been checked, a new SYSCOPY record is written with an ICTYPE of T, an STYPE of F (full), or I (incremental), and a TTYPE of B (for broken page) for the table space or index space. This entry prevents subsequent incremental image copies from being taken. Incremental copies make no sense at this point because the previous full copy that encountered a check-page error has not copied the full table space to the image copy. Furthermore, the pages that were already copied are marked as *not changed*. Therefore, a subsequent incremental copy would rely on these pages already being available in a valid backup, but they are not available in a backup. Therefore, the need to suppress incremental copies.

Be sure to identify COPY utilities that terminate with a return code of 8 and fix the underlying problem as soon as practical.

## SCOPE PENDING for COPY

The COPY utility has been enhanced to optionally copy only objects that are in copy pending or informational copy pending state by specifying the SCOPE PENDING option. See Example 3-16. This option can be powerful when used with the LISTDEF support.

**Note:** When DSNUM ALL is specified, the entire table space or index space is copied if one or more partitions are in copy or information copy states. To copy just those parts that are copy or informational copy pending, then this can be achieved using LISTDEF and the PARTLEVEL option

*Example 3-16 Example of a COPY with TEMPLATE*

---

```
TEMPLATE LOCALDDN UNIT SYSDA DSN(COPY001F.IFDY01)
          SPACE(15,1) CYL DISP(NEW,CATLG,CATLG)
COPY TABLESPACE DSN9D81A.DSN9S91E COPYDDN(LOCALDDN)
      SHRLEVEL REFERENCE SCOPE PENDING
```

---

## TEMPLATE switching

The new template switching function allows image copies of varying sizes to have different characteristics. This function provides significant flexibility in terms of the data set names and attributes, for example, device types.

Template switching is available for image copies produced by the following utilities:

- ▶ COPY
  - FULL YES/NO
  - CONCURRENT
- ▶ COPYTOCOPY
- ▶ MERGECOPY
- ▶ LOAD
- ▶ REORG

Both COPYDDN and RECOVERYDDN utility options can support the switching of templates. Template switching is controlled by the new LIMIT template keyword. The LIMIT keyword has two mandatory operands:

- ▶ The maximum primary allocation that is permitted for the template
- ▶ The new template to be used when this maximum is reached

Note that you can switch templates only once.

In Example 3-17 on page 155, if MY.SMALLTS table space is 20 cylinders in size, it can be accommodated within the small template, specified by the COPYDDN keyword. The image copy data set is allocated on disk. If MY.LARGETS table space is 1000 cylinders, it exceeds the limit, specified by the COPYDDN keyword (small), and is switched to the large template. The image copy data set is then allocated on tape.



```
//SYSIN DD *  
TEMPLATE small DSN &DB..&TS..IC.D&DA..T&TI. UNIT=DASD LIMIT(100 CYL,large)  
TEMPLATE large DSN &DB..&TS..IC.D&DA..T&TI. UNIT=TAPE  
COPY TABLESPACE MY.SMALLTS COPYDDN(small)  
COPY TABLESPACE MY.LARGETS COPYDDN(small)
```

---

### **COPYTOCOPY option**

The COPY utility and the LOAD and REORG utilities with *Inline COPY* can produce local primary and backup copies and recovery site primary and backup copies. Taking two local and two recovery sites copies all at once can tie up four tape drives per utility, and if multiple copies are running concurrently, there may be a need for more tape drives than are available to the site. In addition, some customers use remote attached tape drives for their recovery site copies, thus causing the copy to take longer and decreasing data availability because of bandwidth constraints. The requirement, therefore, is to be able to take a single copy and later make additional copies, asynchronously from the original, and record the copies in SYSCOPY.

Input to the utility is either the local primary or the recovery site primary copy from which COPYTOCOPY can make up to three copies of one or more of the following types:

- ▶ Local primary
- ▶ Local backup
- ▶ Recovery site primary
- ▶ Recovery site backup

COPYTOCOPY does not support the following items:

- ▶ DSNDB01.SYSUTILX and its indexes
- ▶ DSNDB01.DBD01 and its indexes
- ▶ DSNDB06.SYSCOPY and its indexes
- ▶ Image copies taken with CONCURRENT option
- ▶ Image copies taken by REORG part range

COPYTOCOPY does not check recoverability of an object.

Use COPYTOCOPY in the following conditions:

- ▶ When additional copies of objects that are high availability are required and the extra copies are taken to slower devices.
- ▶ When the target object is required to remain in UTRW status, allowing SQL statements to run concurrently with the same target object.

The COPYTOCOPY option can also use the LISTDEF and TEMPLATE functionality as described; see Example 3-18 on page 156. This example also illustrates the statements that are required to take three copies from the local primary copy. Using COPYTOCOPY to take a copy of the local primary and creating another local primary copy is not possible.

*Example 3-18 COPYTOCOPY example with TEMPLATE usage*

```
TEMPLATE LOCALDDN
    DSN(DB2V910G.&DB..&TS..D&DATE..T&TIME.&LOCREM.)
    VOLUMES(SBOX60)
TEMPLATE RECOVDDN
    DSN(DB2V910G.&DB..&TS..D&DATE..T&TIME.&LOCREM.)
    VOLUMES(SBOX61)
TEMPLATE RECOVDD2
    DSN(DB2V910G.&DB..&TS..D&DATE..T&TIME.Z)
    VOLUMES(SBOX62)
LISTDEF DB01A
    INCLUDE TABLESPACE U9G01T11.TSPSUPP1
    COPYTOCOPY LIST DB01A FROMLASTCOPY
COPYDDN(,LOCALDDN)
RECOVERYDDN(RECOVDDN,RECOVDD2)
```

**Note:** The **TIMESTAMP** field in **SYSCOPY** is the timestamp of the original image copy. If date and time are used in the data set name, as in Example 3-18, these values will contain the date and time that the **COPYTOCOPY** was run and not the date and time in the input copy data set

## CLONE option

With the **CLONE** option, the **COPY** utility copies only a **CLONE TABLE** or **INDEX** data. If you use **LIST** keyword to specify a list of objects, **COPY** processes only clone table or indexes on clone tables. Example 3-19 shows how **COPY** works with the **CLONE** option. There are two table spaces to be copied but only one has a clone table. Then, the processing skipped message appears.

*Example 3-19 COPY using CLONE option*

```
//COPY1 EXEC DSNUPROC,SYSTEM=DB9A,UID='BROONEY'
//DSNUPROC.SYSIN DD *
TEMPLATE COPY_TS DSN &DB..&TS..&IC.D&DA..T&TI.
    UNIT=SYSDA
LISTDEF COMPLIST
    INCLUDE TABLESPACE DB1.TS1
    INCLUDE TABLESPACE DB2PEDB.DB2PMFAC
COPY LIST COMPLIST COPYDDN(COPY_TS) CLONE

12:29:42.35 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = DB2R7.DB2R7001
12:29:42.39 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
12:29:42.39 DSNUGUTC - TEMPLATE COPY_TS DSN &DB..&TS..&IC.D&DA..T&TI. UNIT=SYSD
12:29:42.39 DSNUJTDR - TEMPLATE STATEMENT PROCESSED SUCCESSFULLY
12:29:42.40 DSNUGUTC - LISTDEF COMPLIST INCLUDE TABLESPACE DB1.TS1 INCLUDE TABLE

12:29:42.40 DSNUIldr - LISTDEF STATEMENT PROCESSED SUCCESSFULLY
12:29:42.40 DSNUGUTC - COPY LIST COMPLIST COPYDDN(COPY_TS) CLONE
12:29:42.41 DSNUGULM - PROCESSING LIST ITEM: TABLESPACE DB1.TS1
12:29:42.41 DSNUGULM - PROCESSING SKIPPED FOR TABLESPACE DB2PEDB.DB2PMFAC REASON CODE=1
12:29:42.46 DSNUGDYN - DATASET ALLOCATED. TEMPLATE=COPY_TS
    DDNAME=SYS00001
    DSN=DB1.TS1.FD01.T163005
12:29:42.51 DSNUBBID - COPY PROCESSED FOR TABLESPACE DB1.TS1
    NUMBER OF PAGES=3
    AVERAGE PERCENT FREE SPACE PER PAGE = 32.33
    PERCENT OF CHANGED PAGES = 1.66
    PERCENT OF CHANGED PAGES = 1.66
```

```

ELAPSED TIME=00:00:00
274 12:29:42.52 DSNUBAFI - DB2 IMAGE COPY SUCCESSFUL FOR TABLESPACE DB1.TS1
12:29:42.53 DSNUGULM - PROCESSING SKIPPED FOR 1 OF 2 OBJECTS
12:29:42.53 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0

```

---

Another option for image copies is the use of the concurrent copy feature, with or without SnapShot.

This option of the COPY utility enables you to make full image copies using DFSMS concurrent COPY. The copy process is initiated by the DB2 COPY utility when you specify the CONCURRENT keyword on the COPY statement. The image copy is recorded in SYSCOPY with ICTYPE = F and STYPE = C for I0001 instance node, and STYPE = J for J0001 instance node.

To use COPY to take DFSMS concurrent copies, the following hardware must be installed:

- ▶ 3990 Model 3 or 3990 Model 6 controller at the extended platform attached to the disk  
A COPY job fails if one or more of the table spaces are on a disk that does not have the right storage controller.
- ▶ Any DASD that supports FlashCopy capability

**Restrictions:**

- ▶ You cannot use a copy made with DFSMS concurrent COPY with the PAGE or *ERRORRANGE* options of the RECOVER utility.
- ▶ You cannot run the following DB2 stand-alone utilities on copies made by DFSMS Concurrent COPY: DSN1COMP, DSN1COPY, DSN1PRNT  
You can manually restore the DFSMS copy data set under a different name and run these utilities against the restored data set.

Concurrent Copy features allow DB2 to create full image copies with only a short time interval of data unavailability. The DB2 RECOVER utility is able to handle these copies. Table 3-6 shows the options available for an image copy with and without the concurrent option.

*Table 3-6 DB2 Image Copy with and without concurrent COPY*

Concurrent	Type		SHRLEVEL	
	FULL	INCR	REFERENCE	CHANGE
Yes	Yes	No	Yes	Yes <sup>a, b</sup>
No	Yes	Yes	Yes	Yes

a. Short unavailability at data set level

b. Not valid for page size larger than 4 KB

Example 3-20 on page 158 shows the error messages you get if you are using concurrent copy option without the appropriated hardware and software support.

*Example 3-20 CONCURRENT COPY output, with error*

---

```
DSNU050I DSNUGUTC - COPY TABLESPACE U9G01T11.TSPSUPP2 COPYDDN(BROONEY) DSNUM 3 CONCURRENT
DSNU1038I DSNUGDYN - DATASET ALLOCATED. TEMPLATE=BROONEY
          DDNAME=SYS00005
          DSN=PAOLOR4.UTIL.TSPSUPP2.P00003.T233225
DSNU421I DSNUBBCM - START OF DFSMS MESSAGES
PAGE 0001 5695-DF175 DFSMSDSS V2R10.0 DATA SET SERVICES 2009.293 19:32
PARALLEL
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PARALLEL'
  DUM OPT(3) DATAS(INCL(PAOLOR1.DSNDBC.U9G01T11.TSPSUPP2.J0001.A003)) -
  CAN CONC SHA TOL(ENQF) WAIT(0,0) -
  OUTDD(SYS00005)
ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'DUM '
ADR109I (R/I)-RI01 (01), 2009.293 19:32:33 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED.
ADR014I (SCH)-DSSU (02), 2009.293 19:32:33 ALL PREVIOUSLY SCHEDULED TASKS COMPLETED. PARALLEL
MODE
NOW IN EFFECT
ADR050I (002)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (002)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (002)-STEND(01), 2009.293 19:32:33 EXECUTION BEGINS
ADR411W (002)-DTDSC(04), DATA SET PAOLOR1.DSNDBC.U9G01T11.TSPSUPP2.J0001.A003 IN CATALOG
UCAT.VSBOX01 ON VOLUME SBOX58
          WAS NOT SERIALIZED ON REQUEST
ADR356E (002)-UIMXT(01), TASK TERMINATED BY UIM EXIT (24)
ADR735W (002)-TOMI (05), 2009.293 19:32:33 USE OF CONCURRENT COPY FAILED FOR DATA SET
          PAOLOR1.DSNDBC.U9G01T11.TSPSUPP2.J0001.A003 ON VOLUME SBOX59, 002.
SERIALIZATION WILL BE HELD AND CONCURRENT COPY WILL NOT BE USED.
ADR356E (002)-UIMXT(01), TASK TERMINATED BY UIM EXIT (23)
ADR801I (002)-DTDSC(01), DATA SET FILTERING IS COMPLETE. 1 OF 1 DATA SETS WERE SELECTED: 0
FAILED
SERIALIZATION AND 0
FAILED FOR OTHER REASONS.
ADR006I (002)-STEND(02), 2009.293 19:32:33 EXECUTION ENDS
ADR013I (002)-CLTSK(01), 2009.293 19:32:33 TASK COMPLETED WITH RETURN CODE 0008
ADR012I (SCH)-DSSU (01), 2009.293 19:32:33 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS
0008
FROM:
          TASK 002
DSNU422I DSNUBBCM - END OF DFSMS MESSAGE
DSNU409I DSNUBBUM - NO HARDWARE SUPPORT FOR TABLESPACE U7G01T11.TSPSUPP2 DSNUM 3
DSNU012I DSNUGBAC - UTILITY EXECUTION TERMINATED, HIGHEST RETURN CODE=8
```

---

## Copying INDEXES

DB2 has the ability to take full image copy of indexes. These copies can then be used by the RECOVER INDEX utility. The RECOVER utility recovers an index in the same manner as a table space, that is, restoring an image copy and then applying the log.

This method allows for faster recovering of an index, assuming timely image copies are taken, when compared to rebuilding the index, and also it allows for the recovery of table spaces and indexes in parallel, thus reducing the outage required.

To allow indexes to be copied, the COPY YES option is available on the CREATE/ALTER index DDL statements. COPY YES can be specified to indicate that the index can be copied by the COPY utility.

The COPY YES attribute has the following features:

- ▶ Allows DB2 full image copies and concurrent copies to be taken.
- ▶ Allows the use of the RECOVER utility on the index.
- ▶ Enables SYSCOPY recording for utility activity on the index.
- ▶ Enables SYSLGRNX recording for the index.
- ▶ Enables the setting of ICOPY and CHKP pending states for the index.
- ▶ Enables down-level detection on the index.
- ▶ Enables RECOVER from SYSTEM LEVEL BACKUP

Altering the index to COPY NO prohibits and disables the attributes and also removes the recovery information from SYSCOPY and SYSLGRNX for the index.

The following restrictions apply to copying an index:

- ▶ CHANGELIMIT cannot be used.
- ▶ The copy data set is a sequential data set with a 4 KB LRECL.
- ▶ Inline image copies cannot be used.
- ▶ Incremental image copies cannot be used.
- ▶ DSNUM ALL must be used for NPI.
- ▶ RECOVER from SYSTEM LEVEL BACKUP could not be used.
- ▶ For compressed indexes, COPY creates an image copy of the index that is not compressed

The index state, ICOPY (informational copy pending) indicates that a copy of the index should be taken and is set by the following ways:

- ▶ REBUILD INDEX
- ▶ REORG INDEX
- ▶ REORG TABLESPACE (LOG YES or LOG NO)
- ▶ LOAD TABLE (LOG YES or LOG NO)
- ▶ ALTER to padded
- ▶ ALTER to not padded
- ▶ ALTER add of a key column
- ▶ ALTER of a numeric data type key column

**Tip:** Use the following guidelines for the image copying of indexes:

- ▶ Use image copy on large indexes with low update activity that require high availability.
- ▶ Copy indexes in conjunction with the underlying table space.
- ▶ Recover table spaces and their indexes to the same point-in-time to avoid CHKP being set.

### Image copy failures during recovery

During a recovery, an image copy may fail (for example, because of an I/O error). In this case, RECOVER attempts to use the dual image copy, assuming that such a copy exists. If the copy does not exist or also fails, RECOVER ignores the copy if it is an incremental image copy, and uses the log for recovery. If the failing image copy is a full copy, RECOVER falls back to an earlier full image copy to complete the recovery. The fallback has a performance penalty, but it helps to ensure availability.

Because the fallback ensures recoverability, certain installation sites do not generate dual image copies. These sites prefer to run frequent incremental image copies instead.

## Deleting image copies and archive logs

Image copies are required for data integrity. The customer must have procedures to ensure that image copies are deleted only when they are not required anymore. Moreover, because the image copies and the archive logs are used together, the deletion of these data sets must be synchronized. For example, there is no use for an archive log that is older than the oldest image copy unless other types of backups, not only image copies, are also used for recovery.

Image copies and archive logs are recorded in DB2 and optionally cataloged in an ICF catalog. Physical deletion of the data sets removes them from the ICF catalog. This physical deletion must be coordinated with a DB2 cleanup procedure to remove obsolete information in SYSIBM.SYSCOPY. This cleanup is performed with the MODIFY utility.

The deletion of image copy data sets from the MVS catalog and the DB2 catalog must also be synchronized with the deletion of the log data sets from the MVS catalog and from the BSDS.

## BACKUP SYSTEM copies

The BACKUP and RESTORE SYSTEM utilities were added in DB2 V8 and use disk volume FlashCopy backups and copy pool z/OS DFSMSHsm V1R5 constructs. In DB2 9, these utilities were enhanced to use new functions available with z/OS V1R8 DFSMSHsm.

## DB2 and FlashCopy

FlashCopy image is an instantaneous copy of DASD volume taken at a particular time. Keeping several versions is possible if the resource is available and the data can be maintained on disk or tape directly with DB2 9.

DFSMSHsm Fast Replication provides DFSMSHsm management for the use of volume level fast replication. With this capability, a set of storage groups can be defined as a copy pool. The volumes in this pool are processed collectively, by creating, with Fast Replication, backup versions managed by DFSMSHsm.

For a FlashCopy to be taken, first, a relationship is established between the Copy Pool1 (source) and the backup Copy Pool (target). Volumes will be logically associated so that a physical copy of each volume can be made. At the point the relationship is established, the BACKUP SYSTEM or RESTORE SYSTEM is considered logically complete.

For example, when doing a BACKUP SYSTEM operation, the unavailability period for access to the system lasts only until the BACKUP SYSTEM is logically complete, therefore it is fast. After logically complete, a background copy is started so that the target looks like the source did at the time the operation was logically complete. If any applications cause changes to tables, so that tracks on the source volume must be updated before they are copied to the target, the source track with the change is first copied to the target before it is updated. After the copy of each volume is complete the logical relationship can go away.

Recovery can be performed at the volume or user-defined-volumes set level. With DFSMSHsm Fast Replication, the backup and recovery of DB2 storage groups can be managed by DFSMSHsm. Non-disruptive backups can be taken automatically. Recovery is also fast and easy to perform. Copy pools and Fast Replication provide a fast, easy to use backup and recovery solution designed to work specifically with DB2 Version 8 or later.

See the following publications for more information:

- ▶ *Disaster Recovery with DB2 UDB for z/OS*, SG24-6370
- ▶ *DB2 UDB for z/OS Version 8: Everything You Ever Wanted to Know, ... and More*, SG24-6079
- ▶ *DFSMSHsm Fast Replication Technical Guide*, SG24-7069

For a FlashCopy to be taken, first, a relationship is established between the Copy Pool1 (source) and the backup Copy Pool (target). Volumes will be logically associated so that physical copy of each volume can be made. At the point the relationship is established, the BACKUP SYSTEM or RESTORE SYSTEM is considered logically complete.

For example, when doing a BACKUP SYSTEM, the unavailability period for access to the system only lasts until the BACKUP SYSTEM is logically complete, so it is very fast. After logically complete, a background copy is started so that the target will look like the source did at the time the operation was logically complete. If any applications cause changes to tables so that tracks on the source volume must be updated before they are copied to the target, the source track with the change is first copied to the target before it is updated. After the copy of each volume is complete the logical relationship can go away.

**Note:** Use the DFSMSHsm FRBACKUP PREPARE when the environment is set up and every time the environment is changed. This command should be run by the storage administrator.

In preparation for the FlashCopy usage, the storage team must run the DFSMSHsm FRBACKUP PREPARE command to assigns specific target volumes to each source volume for each version of the copy pool.

For detailed information about the DFSMSHsm Fast Replication function and FlashCopy, see *DFSMSHsm Fast Replication Technical Guide*, SG24-7069.

One enhancement allows individual table spaces or index spaces to be recovered (the V8 support was for the entire system). Maintaining multiple copies of all the data on disk can be expensive; DB2 9 also allows for the backup to be implemented directly to tape.

The physical copying to disk in the background can be improved by the use of incremental FlashCopy. Even if incremental FlashCopy is used, the dump to tape is always a full dump. Incremental FlashCopy has no benefit when dumping to tape except that the dumping to tape might begin earlier because less data has be copied to disk before writing to tape can be started.

**Important:** If you are planning to use BACKUP SYSTEM utility, you must have a dedicated DASD for each DB2 subsystem or data-sharing group. If you have table spaces from more than one DB2 in the same DASD volume, when you execute a RESTORE SYSTEM for a specific DB2 subsystem, inconsistent data might be restored.

### 3.9.5 TOKEN

The token is a 36-digit hexadecimal byte string that uniquely identifies each system-level backup and is reported in the PRINT LOG MAP utility. See the DSNJU004 output in Example 3-21 on page 162. The token is stored by DFSMSHsm in its backup control data set (BCDS) records. The DB2 token breakout is shown in Table 3-7.

Table 3-7 DB2 token breakout and description

Field contents	Field description	Length
DB2 SSID	DB2 Subsystem ID	4
TOD	Time of Day of SYSPITR = LRSN (GMT)	8
RBA	Checkpoint RBA	6

Example 3-21 BSDS BACKUP SYSTEM utility history output from DSNJU004

BACKUP SYSTEM UTILITY HISTORY						
SUBSYSTEM ID D9C1						
20:11:08 MAY 9, 2010						
START STCK	DATA COMPLETE		DATA/LOG COMPLETE			
DATA	LOG	RBLP	LRSN	DATE	LTIME	LOCATION NAME
C1EBB66942D63C09	C1EBB66C54A45D43	C1EA802CE63B	C1EBB675C2D2	2010/05/09	15:06:02	DB9C
TOKEN = C4F9C3F1C1EBB66942D63C09C1EA802CE63B						
C1EA8EA7799EE44E	C1EA8EA9FB0E8443	C1EA802CE63B	C1EA8EB4F110	2010/05/09	17:02:51	DB9C
TOKEN = C4F9C3F1C1EA8EA7799EE44EC1EA802CE63B						

### 3.9.6 DFSMS COPYPOOL

SMS was enhanced in z/OS DFSMS V1.5 to support the COPYPOOL function. The construct named copy pool and the copy pool backup storage group type were introduced. You may use the copy pool construct to define which storage group should be processed for fast replication functions. The copy pool backup storage group type is used to define which volumes DFSMSHsm may use as the target volumes of the fast replication backup versions.

A *copy pool* is a set of SMS pool storage groups that can be processed by fast replication operations as one unit with one command. The Interactive Storage Management Facility (ISMF), which is the menu-driven application used to control SMS, has been enhanced to support the SMS enhancements.

A copy pool can contain up to 256 pool storage groups to be processed for fast replication operations, and each pool storage group must be associated with a new type of storage group called the copy pool backup storage group. A pool storage group can have only one associated copy pool backup storage group, and many pool storage groups can be associated with the same copy pool backup storage group. Therefore, in a copy pool backup storage group, you can have different versions of different pool storage groups all together.

HSM keeps the inventory of the backup versions in the BCDS. The BSDS requests a version by specifying a token that is associated with the needed RBA. Each copy pool has a VERSIONS attribute that specifies how many versions should be maintained on disk, with a default of 2 and a maximum of 85.

Volumes to be copied are evaluated at processing-time rather than at definition-time so that changes to the copy pool after definition are reflected in future processing. The copy pool backup storage group must contain enough volumes for a unique one-to-one relationship with the volumes in the pool storage group.

#### Restrictions:

- ▶ FlashCopy V2: An eligible target volume must have the same track form as the source volume, be the exact size of the source volume, and reside in the same Enterprise Storage Server (ESS) as the source volume.
- ▶ FlashCopy V1: The same restrictions apply as for FlashCopy V2. In addition, the target and the source volumes must reside in the same LSS.

In summary, a *copy pool* is a set of SMS storage groups, which contain data that DFSMSHsm can back up collectively. After the backup is taken, it is identified by a token.

**Warning:** The data within a copy pool must not be migrated.



### 3.9.7 DUMPCLASS

Use DUMPCLASS option when you want to dump to tape. It indicates the DFSMSHsm dump class that you want to use for the dump processing. You can specify up to five dump classes. If you do not specify a dump class, DB2 uses the default dump classes that are defined for the copy pools.

### 3.9.8 Object-level backups

The backups taken by the BACKUP SYSTEM utility (referred to here as system level backups) in V8 are used for recovery of the entire DB2 system. DB2 9 has an enhancement to allow a subset of the data to be recovered from the system-level backups. Recovery is through the RECOVER utility, which is now capable of using system-level backups for the restore phase in addition to image copies. To allow the RECOVER utility to consider using system level backups, a DSNZPARM option is available:

`DSN6SPRM.SYSTEM_LEVEL_BACKUPS`

This option can be set from panel DSNTIP6. After enabling the use of system-level backups, they will be automatically considered in object level recoveries.

**Note:** You must alter your indexes to the COPY YES attribute to enable the RECOVER utility from system-level backups.

If the installation site is also planning to offload the system-level volume backups to tape, see 3.10.1, “BACKUP SYSTEM utility” on page 165; there are no syntax changes to RECOVER. The user specifies the table spaces or indexes in a list, individually, or through LISTDEF, and the RECOVER utility will determine the most recent recovery base. In addition to the image copies recorded in the SYSCOPY (or the log for some catalog or directory spaces), the system-level volume backups are also examined. If the last copy before the recovery point is a system-level volume backup, the specific data sets are extracted and restored with DFSMSHsm.

Unless the site is also planning to offload the system-level volume backups to tape, there are no syntax changes to RECOVER. The user specifies the table spaces or indexes in a list, individually, or with a LISTDEF, and the RECOVER utility will determine the most recent recovery base. In addition to the image copies recorded in the SYSCOPY (or the log for some catalog or directory spaces), the system-level volume backups are also examined. If the last copy before the recovery point is a system-level volume backup, then the specific data sets will be extracted and restored with DFSMSHsm.

**Restriction:** If a data set has moved to a separate volume or was deleted between the time of the BACKUP SYSTEM utility and the time of the RECOVER utility, then object level recovery is not possible unless an image copy is available.

Allowing system-level volume backups to be used by RECOVER means that conventional image copies must be taken on a potentially much-reduced frequency. For example, if there is a system backup daily, then running image copies daily might not be needed, unless for special reasons such as DR. Because of the restriction noted previously, we do not recommend that image copies are dispensed entirely. Image copies will still be required following LOAD REPLACE and REORG LOG NO, as enforced by DB2.

**Note:** Running BACKUP SYSTEM now results in Real Time Statistics columns for COPY to be updated because system-level backups may now be used in object-level recovery.

In the installation panel, you use the options shown in Figure 3-10 to specify whether the RECOVER utility should use system-level backups as a recovery base (this is in addition to sequential image copies and concurrent copies) for object-level recoveries.

```
DSNTIP6          INSTALL DB2 - DB2 UTILITIES PARAMETERS
====>

Enter system-level backup options for RESTORE SYSTEM and RECOVER below:
1 SYSTEM-LEVEL BACKUPS  ====> NO           As a recovery base: NO or YES
2 RESTORE/RECOVER      ====> NO           From dump: NO or YES
3 DUMP CLASS NAME      ====> PAOLOD      For RESTORE/RECOVER from dump
4 MAXIMUM TAPE UNITS   ====> NOLIMIT    For RESTORE SYSTEM: NOLIMIT or 1-255

Enter other DB2 Utilities options below:
5 TEMP DS UNIT NAME    ====> SYSDA      Device for temporary utility data sets
6 UTILITY CACHE OPTION ====> NO         3990 storage for DB2 utility IO
7 STATISTICS HISTORY   ====> NONE      Default for collection of stats history
8 STATISTICS ROLLUP    ====> NO         Allow statistics aggregation: NO or YES
9 STATISTICS CLUSTERING====> ENHANCED   For RUNSTATS (ENHANCED or STANDARD)
10 UTILITY TIMEOUT     ====> 6         Utility wait time multiplier
11 UT SORT DS ALLOCATION====> YES        Predictable sort disk space allocation
12 IGNORE SORTNUM STMT  ====> YES        Ignore SORTNUM keyword in UT stmt
```

Figure 3-10 DSNTIP6 installation panel: specifying system-level backups

### SYSTEM-LEVEL BACKUPS

Use this option to specify whether the RECOVER utility should use system-level backups as a recovery base (in *addition* to image copies and concurrent copies) for object-level recoveries).

This is where you specify the name of the DFSMSHsm dump class that will be used by the RESTORE SYSTEM utility to restore from a system-level backup that has been dumped to tape.

This is also the dump class that is used by the RECOVER utility to restore objects from a system-level backup that has been dumped to tape. The setting is applicable only when you specify YES in field 2, RESTORE/RECOVER. You can override the setting for DUMP CLASS NAME by executing the RESTORE SYSTEM utility statement or the RECOVER utility statement with the DUMPCLASS keyword.

### Other types of copies

DB2 table and index spaces can be copied by other utilities, not under DB2 control. This can include both IBM (DFSMSdfp, DSN1COPY) and products other than IBM products. DB2 has a limited or no support for these copies. The copies must be restored outside of DB2, and the user must execute a RECOVER with option LOGONLY to apply the changes not reflected in the external copy to help maintain data integrity and consistency.

## 3.10 DFSMSHsm and DB2 BACKUP SYSTEM

Starting with DB2 Version 8, DB2 introduced two utilities, BACKUP and RESTORE SYSTEM. These utilities are to help address issues with critical DB2 systems that require high availability with fast and non-intrusive backup facilities and fast recovery capabilities to minimize down time. Typical uses are as follows:

- ▶ Implementing a recovery of an ERP application to a point-in-time
- ▶ Establishing a full system back up to transport to a remote location
- ▶ Cloning the whole DB2 subsystem.

In this section, we provide a brief description of DFSMS terms, an overview of the system-related functions of BACKUP and RESTORE SYSTEM, and the execution of certain scenarios of recovery at the object level. See Figure 3-11. For detailed information and descriptions, see Chapter 4, “System managed DB2 data” on page 193.

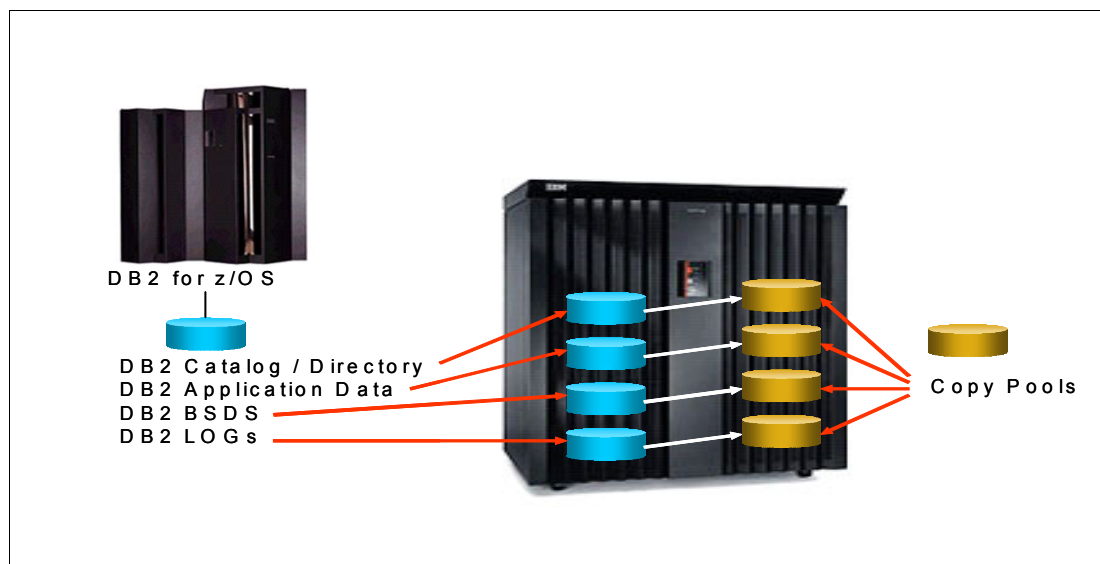


Figure 3-11 Database and storage integration

The BACKUP SYSTEM is based on DFSMSHsm functions and therefore certain DFSMS terms are used in this section. Several of those terms are described in this section.

### 3.10.1 BACKUP SYSTEM utility

BACKUP SYSTEM support was introduced in DB2 for z/OS V8 to provide an easier and less disruptive way for fast volume-level backup and recovery. Its implementation was based on the usage of FlashCopy to backup DB2 data and logs, consequently eliminating the need to suspend logs. System-level backups are managed by DB2 and DFSMSHsm to support system-level point-in-time (PIT) recovery.

BACKUP SYSTEM utility works in an online fashion and allows concurrent applications to continue to access the database for read and write operations. To exploit the BACKUP SYSTEM support, all DB2 data sets and logs must be SMS-managed and be part of DB2 DFSMSHsm copy pools.

BACKUP SYSTEM requires that you define two copy pools for each DB2 system, as follows:

- ▶ A data copy pool, containing all application data and the DB2 catalog and directory and corresponding ICF catalogs
- ▶ A log copy pool containing the DB2 log data sets, BSDS and corresponding ICF catalogs

The BACKUP SYSTEM gives you the option to either copy both copy pools or copy the data copy pool only. If you want to restore the DB2 system or an object to an arbitrary point in time, the option of copying the data copy pool only should be sufficient. Alternatively, if you plan to clone the DB2 at system-level or if you want to restore DB2 to the point when the backup was created, copying both copy pools is required.

These copy pools must adhere to the following name convention:

DSN\$<location-name>\$<cp-type>

The name has the following definitions:

<b>DSN</b>	The unique DB2 product identifier
<b>\$</b>	Fixed character delimiter
<b>&lt;location-name&gt;</b>	DB2 location name
<b>\$</b>	Fixed character delimiter
<b>&lt;cp-type&gt;</b>	Copy Pool type

Use DB for data copy pool and LG for log copy pool, for example:

DATA COPYPOOL (DSN\$DB9A\$**DB**)

LOG COPYPOOL (DSN\$DB9A\$**LG**)

In a data sharing environment, if any failed or abnormally quiesced members exist, the BACKUP SYSTEM request fails. The BACKUP SYSTEM utility uses copy pools, which are constructs in z/OS DFSMSHsm V1R5 and later. Each DB2 subsystem must have two copy pools, one for databases and one for logs.

BACKUP SYSTEM copies the volumes that are associated with these copy pools at the time of the copy. The output for BACKUP SYSTEM is the copy of the volumes on which the DB2 data and log information resides. The BACKUP SYSTEM history is recorded in the bootstrap data sets (BSDSs) in the BACKUP SYSTEM UTILITY HISTORY section. Up to 50 backup versions can be stored

**Note:** In a data sharing environment, the submitting member records the version in its BSDS and also in the shared communications area (SCA) in the coupling facility. If you want to see the latest backup system entry, run DSNJU004 with the GROUP DD statement pointing to the BSDS data sets. This process merges all the BSDS information from all the members.

In the following examples, we show a DB2 BACKUP SYSTEM FULL job and its related actions as seen from the FAST REPLICATION commands and also the output in DFSMSHsm.

You execute the BACKUP SYSTEM (DB2 Utility) as a stand-alone job. Example 3-22 on page 167 shows the JCL for BACKUP SYSTEM. For additional information about the utilities and the related options, see the *DB2 Version 9.1 for z/OS Utility Guide and Reference*, SC18-9855.

### Example 3-22 BACKUP SYSTEM JCL

---

```
//BACKUP EXEC DSNUPROC,SYSTEM=D9CG,
//          UID='SYSBACK'
//STEPLIB DD DISP=SHR,DSN=DB9C9.SDSNEXIT
//          DD DISP=SHR,DSN=DB9C9.SDSNLOAD
//SYSIN   DD *
          BACKUP SYSTEM FULL
```

---

In Example 3-23, the output from a DB2 BACKUP SYSTEM utility job denotes the backup of the DB COPYPOOL (DSN\$DB9C\$DB).

### Example 3-23 FAST REPLICATION output in DFSMSHsm for the DB COPYPOOL

---

```
ARC1801I FAST REPLICATION BACKUP IS STARTING FOR COPY POOL DSN$DB9C$DB, AT 17:07:44 ON 2008/02/12,
  TOKEN=X'C4F9C3F1C1F0D91FD1AE7903C1F0D58C8ED3'
ARC0640I ARCFRTM - PAGE 0001      5695-DF175 DFSMSDSS V1R09.0 DATA SET SERVICES      2008.043 17:07
ARC0640I ARCFRTM - ADR035I (SCH)-PRIME(06), INSTALLATION EXIT ALTERED BYPASS FAC CLASS CHK DEFAULT TO YES
ARC0640I ARCFRTM - PARALLEL
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PARALLEL'
ARC0640I ARCFRTM - COPY IDY(SBOX5S) ODY(SBOX6A) DUMPCOND FR(REQ) PUR ALLX ALLD(*)
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'COPY'
...
...
ARC0640I ARCFRTM - ADR012I (SCH)-DSSU (01), 2008.043 17:07:45 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000
ARC1805I THE FOLLOWING 00004 VOLUME(S) WERE SUCCESSFULLY PROCESSED BY FAST REPLICATION BACKUP OF COPY POOL DSN$DB9C$DB
ARC1805I (CONT.) SBOX5S
ARC1805I (CONT.) SBOX5T
ARC1805I (CONT.) SBOX5U
ARC1805I (CONT.) SBOX5V
ARC1802I FAST REPLICATION BACKUP HAS COMPLETED FOR COPY POOL DSN$DB9C$DB, AT 17:07:45 ON 2008/02/12, FUNCTION RC=0000,
  MAXIMUM VOLUME RC=0000
```

---

Next, Example 3-24 shows the output of the FAST REPLICATION (as invoked by the DB2 BACKUP SYSTEM) for the LOG COPYPOOL (DSN\$DB9C\$LG).

### Example 3-24 FAST REPLICATION output in DFSMSHsm for the LG COPYPOOL

---

```
ARC1801I FAST REPLICATION BACKUP IS STARTING FOR COPY POOL DSN$DB9C$LG, AT 17:07:45 ON 2008/02/12,
  TOKEN=X'C4F9C3F1C1F0D91FD1AE7903C1F0D58C8ED3'
ARC0640I ARCFRTM - PAGE 0001      5695-DF175 DFSMSDSS V1R09.0 DATA SET SERVICES      2008.043 17:07
ARC0640I ARCFRTM - ADR035I (SCH)-PRIME(06), INSTALLATION EXIT ALTERED BYPASS FAC CLASS CHK DEFAULT TO YES
ARC0640I ARCFRTM - PARALLEL
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PARALLEL'
ARC0640I ARCFRTM - COPY IDY(SBOX5Q) ODY(SBOX6G) DUMPCOND FR(REQ) PUR ALLX ALLD(*)
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'COPY'
..
..
ARC0640I ARCFRTM - ADR241I (003)-DDTFP(01), TARGET VTOC BEGINNING AT 000005:0000 AND ENDING AT 000016:0014 IS OVERLAID
ARC0640I ARCFRTM - ADR806I (003)-TOMI (02), VOLUME SBOX5R WAS COPIED USING A FAST REPLICATION FUNCTION
ARC0640I ARCFRTM - ADR006I (003)-STEND(02), 2008.043 17:07:46 EXECUTION ENDS
ARC0640I ARCFRTM - ADR013I (003)-CLTSK(01), 2008.043 17:07:54 TASK COMPLETED WITH RETURN CODE 0000
ARC0640I ARCFRTM - ADR006I (019)-STEND(01), 2008.043 17:07:46 EXECUTION BEGINS
..
..
ARC1805I (CONT.) SBOX5L
ARC1805I (CONT.) SBOX5M
ARC1805I (CONT.) SBOX5N
ARC1805I (CONT.) SBOX5O
ARC1805I (CONT.) SBOX5P
ARC1802I FAST REPLICATION BACKUP HAS COMPLETED FOR COPY POOL DSN$DB9C$LG, AT 17:07:55 ON 2008/02/12, FUNCTION RC=0000,
  MAXIMUM VOLUME RC=0000
```

---

### 3.10.2 RESTORE SYSTEM utility

The DB2 RESTORE SYSTEM utility can be used when you want to recover a subsystem or data sharing group to an arbitrary point-in-time. The utility restores only the database copy pool of a data only or full system backup, and then applies logs until it reaches a place in the log equal to the log truncation point that was specified in a point-in-time conditional restart control record (SYSPITR CRCR), created with DSNJU003. You cannot explicitly name the backup to use for recovery. That is implicitly determined by the log truncation point used to create the SYSPITR CRCR. You can run the DB2 DSNJU004 utility to list the contents of the BSDS as shown in Example 3-25.

*Example 3-25 BSDS BACKUP SYSTEM UTILITY HISTORY output from DSNJU004*

---

BACKUP SYSTEM UTILITY HISTORY						
SUBSYSTEM ID D9C1						
20:11:08 FEBRUARY 11, 2008						
START STCK	LOG	RBLP	DATA COMPLETE	DATA/LOG COMPLETE		
DATA	LOG	RBLP	LRSN	DATE	LTIME	LOCATION NAME
-----	-----	-----	-----	-----	-----	-----
C1EBB66942D63C09	C1EBB66C54A45D43	C1EA802CE63B	C1EBB675C2D2	2008/02/08	15:06:02	DB9C
	TOKEN = C4F9C3F1C1EBB66942D63C09C1EA802CE63B					
C1EA8EA7799EE44E	C1EA8EA9FB0E8443	C1EA802CE63B	C1EA8EB4F110	2008/02/07	17:02:51	DB9C
	TOKEN = C4F9C3F1C1EA8EA7799EE44EC1EA802CE63B					

---

Similar to the Recovery utility, RESTORE can optionally use the Fast Log Apply (FLA) option. The RESTORE SYSTEM utility uses the Recovery Based Log Point (RBLP) stored in the header page of DBD01 and updated by the BACKUP SYSTEM utility as the log scan starting point. The log-apply phase uses Fast Log Apply to recover objects in parallel. DB2 handles table space and index space creates, drops and extends, and marks objects that have had LOG NO events as RECP (table spaces and indices with the COPY YES attribute) or RBDP (indices with COPY NO attribute). An informational message is issued to let the user know whether any objects need additional recovery.

If you want to restore a system to the point at which a backup was taken, do not use the RESTORE SYSTEM utility. Use the following DFSMSHsm command to restore both database and log copy pools:

```
FRRECOV COPYPOOL(cpname) GEN(gen)
```

Then, start DB2 which will use normal restart recovery processing to back out inflight URs.

You cannot specify a backup version with the RESTORE SYSTEM utility. RESTORE SYSTEM utility uses the latest version before the log truncation point. You can specify the log truncation point with the CRESTART SYSPITR option of the DSNJU003 stand-alone utility. See Example 3-26.

*Example 3-26 Create a system restart record using DSNJU003 JCL*

---

```
//D9C1CRCR EXEC PGM=DSNJU003
//*
//STEPLIB DD DISP=SHR,DSN=DB9C9.SDSNEXIT
// DD DISP=SHR,DSN=DB9C9.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=DB9CL.D9C1.BSDS01
//SYSUT2 DD DISP=SHR,DSN=DB9CL.D9C1.BSDS02
//SYSIN DD *
CRESTART CREATE,SYSPITR=C1F0D92A16DB,FORWARD=YES,BACKOUT=YES
```

---

Example 3-27 shows the output of the DSNJU003 job, which was used to create a SYSPITR record in the BSDS.

*Example 3-27 Output of DSNJU003 to create a SYSPITR record in the BSDS*

---

```
CRESTART CREATE,SYSPITR=C1F0D92A16DB,FORWARD=YES,BACKOUT=YES
DSNJ408I DSNRJFCK CHECKPOINT RBA FOUND, RBA = 00000E114090, TIME = 21:51:45 FEBRUARY 12, 2008
DSNJ411I DSNRJRCR CRESTART CREATE FOR CRCRID = 0003, DDNAME = SYSUT1
DSNJ408I DSNRJFCK CHECKPOINT RBA FOUND, RBA = 00000E114090, TIME = 21:51:45 FEBRUARY 12, 2008
DSNJ411I DSNRJRCR CRESTART CREATE FOR CRCRID = 0003, DDNAME = SYSUT2
DSNJ225I CRESTART OPERATION COMPLETED SUCCESSFULLY
DSNJ200I DSNJU003 CHANGE LOG INVENTORY UTILITY PROCESSING COMPLETED SUCCESSFULLY
```

---

Example 3-28 shows the RESTORE SYSTEM utility.

*Example 3-28 RESTORE SYSTEM sample JCL*

---

```
//STEP1 EXEC DSNUPROC,SYSTEM=DB9C,UID=BROONEY
//* UTSTATS=''
//SYSIN DD *
  RESTORE SYSTEM
```

---

Example 3-29 is the output of the DB2 RESTORE SYSTEM job.

*Example 3-29 RESTORE SYSTEM job output*

---

```
DSNU000I DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = BROONEY
DSNU1044I DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNU050I DSNUGUTC - RESTORE SYSTEM
DSNU1606I DSNUVBRD - RESTORE SYSTEM UTILITY STARTING,
COPYPOOL = DSN$DB9C$DB
TOKEN = X'D7F8F7F0BA140298CDE3D14200120CDAC090'.
DSNU1627I DSNUVBRD - RESTORE SYSTEM PRE-LOG APPLY COMPLETED SUCCESSFULLY,
COPYPOOL = DSN$DB9C$DB
TOKEN = X'D7F8F7F0BA140298CDE3D14200120CDAC090'
ELAPSED TIME = 00:00:04.
DSNU1604I -P870 DSNUVARL - RESTORE SYSTEM PHASE LOG APPLY STARTED AT LOG POINT = X'00120CDAC090'.
DSNU1628I DSNUVBRD - RESTORE SYSTEM PHASE LOG APPLY COMPLETED, ELAPSED TIME = 00:04:54.
DSNU010I DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
```

---

In Example 3-30, the DFSMSHsm messages denote the output from the fast replication recovery. This output is the result of the DB2 RESTORE SYSTEM utility, which invoked DFSMSHsms to perform the volume restorations.

*Example 3-30 Recovery messages in DFSMSHsm log*

---

```
ARC1801I FAST REPLICATION RECOVERY IS STARTING FOR COPY POOL DSN$DB9C$DB, AT 21:37:41 ON
2008/02/12
ARC0640I ARCFRTM - PAGE 0001 5695-DF175 DFSMSDSS V1R09.0 DATA SET SERVICES 2008.043 21:37
ARC0640I ARCFRTM - ADR035I (SCH)-PRIME(06), INSTALLATION EXIT ALTERED BYPASS FAC CLASS CHK
DEFAULT TO YES
ARC0640I ARCFRTM - PARALLEL
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PARALLEL'
ARC0640I ARCFRTM - COPY IDY(SBOX6A) ODY(SBOX5S) DUMPCOND FR(REQ) PUR ALLX ALLD(*)
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'COPY '
ARC0640I ARCFRTM - COPY IDY(SBOX6B) ODY(SBOX5T) DUMPCOND FR(REQ) PUR ALLX ALLD(*)
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 003 HAS BEEN ASSIGNED TO COMMAND 'COPY '
ARC0640I ARCFRTM - COPY IDY(SBOX6C) ODY(SBOX5U) DUMPCOND FR(REQ) PUR ALLX ALLD(*)
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 004 HAS BEEN ASSIGNED TO COMMAND 'COPY '
```

```

ARC0640I ARCFRTM - COPY IDY(SBOX6D) ODY(SBOX5V) DUMPCOND FR(REQ) PUR ALLX ALLD(*)
ARC0640I ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 005 HAS BEEN ASSIGNED TO COMMAND 'COPY '
ARC0640I ARCFRTM - ADR109I (R/I)-RI01 (01), 2008.043 21:37:41 INITIAL SCAN OF USER CONTROL
STATEMENTS COMPLETED
ARC0640I ARCFRTM - ADR014I (SCH)-DSSU (02),
2008.043 21:37:41 ALL PREVIOUSLY SCHEDULED TASKS COMPLETED. PARALLEL MODE NOW IN EFFECT
..
..
ARC0401I LARGEST EXTENTS FOR SBOX5V ARE CYLINDERS 9214, TRACKS 138210
ARC0402I VTOC FOR SBOX5V IS 0180 TRACKS(09000 DSCBS), 08836 FREE DSCBS(98% OF TOTAL)
ARC1805I THE FOLLOWING 00004 VOLUME(S) WERE SUCCESSFULLY PROCESSED BY FAST REPLICATION RECOVERY
OF COPY POOL DSN$DB9C$DB
ARC1805I (CONT.) SBOX5S
ARC1805I (CONT.) SBOX5T
ARC1805I (CONT.) SBOX5U
ARC1805I (CONT.) SBOX5V
ARC1802I FAST REPLICATION RECOVERY HAS COMPLETED FOR COPY POOL DSN$DB9C$DB, AT 21:37:42 ON
2008/02/12, FUNCTION RC=0000,
..

```

---

For additional information about SYSTEM BACKUP and RESTORE, and the IBM Recovery Expert for z/OS tool, see the following publications:

- ▶ *Disaster Recovery with DB2 UDB for z/OS*, SG24-6370
- ▶ *Optimizing Restore and Recovery Solutions with DB2 Recovery Expert for z/OS V2.1*, SG24-7606

## 3.11 Online CHECK DATA

When integrity issues are suspected, the CHECK DATA utility can be run to validate the table or tables concerned. CHECK DATA looks at table check constraint, referential constraints, or compare LOB table spaces with the base table space. In DB2 V8, CHECK DATA reduced the access to the table or table spaces to read-only for the duration of the utility executing. For large tables or complex referential integrity (RI) environments, a significant outage of full access to the data could result.

If CHECK DATA finds a violation, the table space is placed into a CHECK pending state causing all the data to be unavailable although as few as one row might have an issue. CHECK DATA resets the CHECK pending status if it finds no errors, or if the option is selected, all rows that contain violations are copied to exception tables and deleted.

In DB2 9, CHECK DATA is enhanced to allow the new CLONE, LOBERROR, and XMLERROR options:

### ▶ CLONE

Indicates that CHECK DATA is to check the clone table in the specified table space. Because clone tables cannot have referential constraints, the utility checks only constraints for inconsistencies between the clone table data and the corresponding LOB data. If you do not specify CLONE, CHECK DATA operates against only the base table.

### ▶ LOBERROR

Specifies the action that CHECK DATA is to perform when it finds a LOB column check error. LOBERROR should not be specified if AUXERROR is specified. If both are



specified, the keywords must match. LOBERROR is ignored for SCOPE XMLONLY because LOB checking is not being performed

► XMLERROR

Specifies the action that CHECK DATA is to perform when it finds an XML column-check error. XMLERROR must not be specified if AUXERROR is specified. However, if both are specified, the keywords must match. XMLERROR is ignored for SCOPE XMLONLY because LOB checking is not being performed.

In DB2 9, CHECK DATA is also enhanced to run in a SHRLEVEL CHANGE mode. The V8 processing is the default or it can be specified as SHRLEVEL REFERENCE. SHRLEVEL CHANGE CHECK DATA works on a copy of the data and indexes (shadows), in Figure 3-12. The copy is taken by DB2 using the DFSMS ADRDSSU utility. For installations with data set level FlashCopy V2 enabled, this is extremely fast. Without the FlashCopy support the outage is still less with SHRLEVEL CHANGE. The utility must be able to drain the objects ahead of the copy. DRAIN WAIT options are available, similar to REORG.

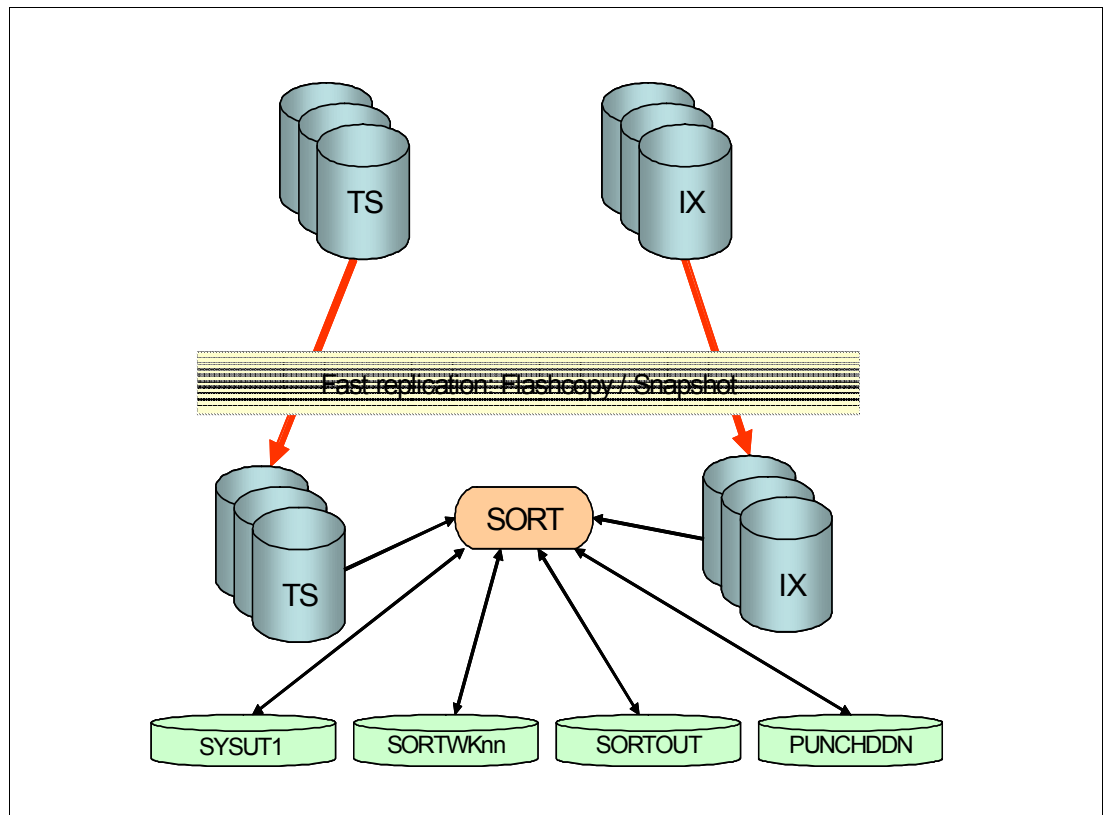


Figure 3-12 DB2 Check data using FlashCopy V2

## 3.12 Other DB2 data sets

Apart from table spaces and recovery data sets, DB2 requires data sets to store the product (libraries) also, to manage its execution (CLISTs, JCL procedures, and work data sets). These data sets are standard z/OS data sets, either partitioned or sequential.

### 3.12.1 DB2 library data sets

DB2 uses a set of library data sets to store distribution code, executable code, ISPF data sets, TSO data sets, SMP/E data sets, and so on. These library data sets are all MVS partitioned data sets (PDS). These data sets are defined during the SMP/E installation of the DB2 product.

Starting in DB2 9, the SDSNLOAD library must be a PDS extended (PDSE). The DB2 SDSNLOAD data set that contains most of the DB2 executable code must now be allocated (with the DSNALLOC job) as a PDSE data set. In DB2 V8, you had the option to convert this data set to a PDSE. If you did not take the opportunity during your DB2 V8 migration to convert this data set to a PDSE, you must now do so. When you allocate the DB2 SDSNLOAD data set as a PDSE, and use it in combination with the IMS Attach, you must apply the IMS toleration PTF UK10146. Also, be aware of the operational differences between PDS and PDS/E data sets. These are explained in the DB2 9 program directory.

**Note:** SMP/E must “know” which type of data set it is managing. It compresses a PDS, but not a PDSE. If you change the data set organization, you must also update the SMP/E definition.

### 3.12.2 DB2 temporary and SORT data sets

DB2 also uses temporary data sets. Examples are utility work and external sort data sets. Most temporary data sets are standard sequential files. These data sets can be explicitly defined in your utility JCL or are created dynamically at utility execution time.

DB2 utilities that invoke DFSORT often experience problems to allocate large-sort work data sets through DFSORT. When utilities are invoking DFSORT, they pass the SORTNUM value as a guidance of how many data sets should be used to allocate the required disk space. In constrained environments, DFSORT might not be able to allocate required disk space in the given number of data sets causing failure of this utility job. See Figure 3-13.

```
DSNTIP6          INSTALL DB2 - DB2 UTILITIES PARAMETERS
====>

Enter system-level backup options for RESTORE SYSTEM and RECOVER below:
 1 SYSTEM-LEVEL BACKUPS ====> NO           As a recovery base: NO or YES
 2 RESTORE/RECOVER      ====> NO           From dump: NO or YES
 3 DUMP CLASS NAME      ====> PAOLOD       For RESTORE/RECOVER from dump
 4 MAXIMUM TAPE UNITS   ====> NOLIMIT      For RESTORE SYSTEM: NOLIMIT or 1-255

Enter other DB2 Utilities options below:
 5 TEMP DS UNIT NAME    ====> SYSDA       Device for temporary utility data sets
 6 UTILITY CACHE OPTION ====> NO          3990 storage for DB2 utility I/O
 7 STATISTICS HISTORY   ====> NONE        Default for collection of stats history
 8 STATISTICS ROLLUP     ====> NO          Allow statistics aggregation: NO or YES
 9 STATISTICS CLUSTERING====> ENHANCED    For RUNSTATS (ENHANCED or STANDARD)
10 UTILITY TIMEOUT      ====> 6           Utility wait time multiplier
11 UT SORT DS ALLOCATION====> YES          Predictable sort disk space allocation
12 IGNORE SORTNUM STMT  ====> YES         Ignore SORTNUM keyword in UT stmt
```

Figure 3-13 Specifying SORT options during installation - Panel DSNTIP6

The DB2 utilities CHECK DATA, CHECK INDEX, CHECK LOB, LOAD, REBUILD INDEX, REORG TABLESPACE, RUNSTATS have been enhanced to allocate sort work data sets dynamically before invoking DFSORT if the new DSNZPARM UTSORTAL is set to YES, and no SORTNUM value is specified in the utility statement. Setting new DSNZPARM IGNSORTN to YES causes utilities to dynamically allocate sort work data sets even if the SORTNUM parameter was specified in the utility statement.

The specified SORTNUM value will then be ignored. When the utility allocates sort work data sets dynamically, it calculates the disk space requirements according to available record estimates and record length. To estimate the number of records to be sorted during utility execution the processed object is looked up in the following real-time statistics tables:

- ▶ SYSIBM.SYSTABLESPACESTATS
- ▶ SYSIBM.SYSINDEXSPACESTATS

If information is not available for the object (that is TOTALROWS or TOTALENTRIES is set to NULL), the current estimation logic based on RUNSTATS catalog statistics will be used.

New DSNZPARMs parameters are UTSORTAL and IFNSORTN (are changeable online):

- ▶ UTSORTAL YES/NO

The UTSORTAL (UT SORT DATA SET ALLOCATION) subsystem parameter specifies whether or not (the default) DB2 is to use real-time statistics to determine the sort work data set sizes and dynamically allocate sort work data sets. If you specify YES and real-time statistics are unavailable, DB2 utilities that invoke sorts (CHECK, LOAD, REBUILD, REORG, and RUNSTATS) use a space prediction algorithm for dynamically allocated sort work data sets.

- ▶ IGNSORTN YES/NO

The IGNSORTN subsystem parameter (IGNORE SORTNUM STAT) determines whether or not (the default) occurrences of the SORTNUM clause in utility control statements are to be ignored. The value of the IGNSORTN is meaningful only when YES is specified in the UT SORT DATA SET ALLOCATION field.

**Note:** But sure to turn on UTSORTAL, test it, and then consider turning on IGNSORTN.

## 3.13 DB2 data sets naming conventions

This section describes the naming standards used by DB2 for its data sets.

### 3.13.1 Table space and index space names

The names for DB2 table spaces and index spaces have the following structure:

h1q.DSNDBx.dbname.spname.ynnnn.Ammm

The elements of this name are as follows:

<b>h1q</b>	VSAM catalog high-level qualifier
<b>DSNDB</b>	Standard part of the name
<b>x</b>	VSAM cluster or data component:
<b>C</b>	Cluster
<b>D</b>	Data

<b>dbname</b>	Database name
<b>spname</b>	Space name of either a table space name or an index name
	Because index names can be more than eight characters long, DB2 sometimes needs to generate an eight-character name. To avoid randomly generated names, and to be able to correlate the index name to the index space, be sure to limit index names to eight characters. This is also true for table names for implicitly defined table spaces (that is, the creation of the table is done without having created the table space), because DB2 will assign a unique table space name.
<b>y</b>	Data set type:
	<i>I</i> Standard data set
	<i>S</i> Shadow data set
	<i>T</i> Temporary data set
<b>nnnn</b>	Number = 0001
<b>A</b>	Standard character, A
<b>mmm</b>	Used for table spaces or index spaces with multiple data sets; <i>mmm</i> is either 001, the data set number, or the partition number.

### 3.13.2 BSDS names

The default names for BSDSs have the following structure:

h1q.BSDS0n

The elements of this name are as follows:

<b>h1q</b>	VSAM catalog high-level qualifier
<b>BSDS0</b>	Standard part of the name
<b>n</b>	BSDS copy, 1 or 2

### 3.13.3 Active log names

The default names for active log data sets have the following structure:

h1q.LOGCOPYn.DSmm

The elements of this name are as follows:

<b>h1q</b>	VSAM catalog high-level qualifier
<b>LOGCOPY</b>	Standard part of the name
<b>n</b>	Active log copy, 1 or 2
<b>DS</b>	Standard part of the name
<b>mm</b>	Active log number, 0 to 92 (after BSDS conversion to the new format)

### 3.13.4 Archive log and BSDS backup names

The default names for archive log and BSDS backup data sets have the following optional structure:

`hlq.ARCHLOGn.Ddyddd.Thhmsst.axxxxx`

The elements of this name are as follows:

<b>hlq</b>	VSAM catalog high-level qualifier
<b>ARCHLOG</b>	Standard part of the name
<b>n</b>	Archive log copy, 1 or 2
<b>Ddyddd</b>	Date, yy=year (2 or 4 digits), ddd=day of year
<b>Thhmsst</b>	Time, hh=hour, mm=minute, ss=seconds, t=tenths
<b>a</b>	A=Archive log, B=BSDS backup
<b>xxxxxx</b>	File sequence

The elements `Ddyddd` and `Thhmsst` are optional qualifiers that are defined in `DSNZPARM DSN6ARVP.TSTAMP` in the `TIMESTAMP ARCHIVES` option (YES or NO) of the `DSNTIPH` panel; and `Ddyddd` can assume the format `Dyyyyddd` if the `TIMESTAMP ARCHIVES` option is set to `EXT` (extended).

### 3.13.5 Image copy names

The names for image copy data sets are not defined by DB2. Each installation must define a standard naming convention to make these data sets distinct and significant. The following sample naming structure is for image copies:

`hlq.wxiiyddd.Thhmmss.ssssss.Ammm`

The elements of this name are as follows:

<b>hlq</b>	VSAM catalog high-level qualifier
<b>w</b>	Copy type, P=Primary, S=Secondary copy
<b>x</b>	Copy requirement, S=Standard, H=Critical
<b>i</b>	Copy frequency, D=Daily, W=Weekly, M=Monthly
<b>yyddd</b>	Date, yy=year, ddd=day of year
<b>Thhmsst</b>	Time, hh=hour, mm=minute, ss=seconds
<b>ssssss</b>	Table space or index space name
<b>Ammm</b>	Data set identifier

### 3.13.6 Determining which DB2 data sets reside on the same disk

Use the following guidelines to determine which DB2 data sets can reside on the same disk:

- ▶ Table spaces and indexes can reside on the same disk if enough PAVs are available. Otherwise separate the table spaces from indexes.
- ▶ DB2 sort data sets (DSNDB07 equivalent) can reside on the same disk with table spaces, indexes, or both, if enough PAVs are available. Otherwise separate the DSNDB07 data sets to their own volumes.

- ▶ DB2 catalog and directory data sets must be placed on their own volumes. You can mix their table spaces and indexes if enough PAVs are available.
- ▶ BSDS and active log data sets can reside on the same volume. Separate the copy 1 and 2 of the BSDS and active log data sets onto different disks, preferably on different disk controllers.
- ▶ Archive log and image copy data sets must be separated from all volumes listed in the previous bullets. Depending on your recovery situation, you might have to separate archive logs from image copy data sets. Otherwise, they can be placed together on the same volume.
- ▶ Most other DB2 data sets, PDSE, PDS, and sequential data sets can typically reside on the same volumes.
- ▶ A good practice is to have data from one subsystem or Data Sharing group reside on separate volumes from other subsystem or Data Sharing groups.

### 3.13.7 DSNZPARMs affecting DB2 data set sizes

Recent synergy between DB2 and SMS allow the two products to be better integrated: DB2 has the ability to include SMS-managed attributes for storage groups. This includes using DATACLAS, MGMTCLAS, and STORCLAS on the CREATE and ALTER STOGROUP commands.

In addition, many sites now use the DB2 space allocations sliding scale for secondary extents to minimize the problem of running out of extents. This is enabled by setting the DSNZPARM MGEXTSZ to YES and an objects SECQTY to -1. DB2 will then manage secondary extent sizes and determine an appropriate secondary space value when an extent is to be allocated.

Use the following guidelines for linear data sets:

- ▶ Allocate sufficient space in cylinders.  
Set PRIQTY and SECQTY to avoid extent processing, particularly for LOAD, REORG, or REBUILD INDEX
- ▶ If secondary extents are unavoidable during utility processing, set DSNZPARM MGEXTSZ to YES to allow DB2 to manage secondary extent allocation and use a high SECQTY or minus 1 (-1).
- ▶ Use DS8000 DASD for best performance and throughput.
- ▶ Use the VARY DS CONTROL INTERVAL parameter set to YES on installation panel DSNTIP7 to allow DB2-managed data sets to have variable VSAM control intervals corresponding to the size of the buffer pool that is used for the table space or index space (page size).
- ▶ Use DB2 and SMS managed data sets  
Use SMS-managed storage groups with a volume identification of \* (asterisk) instead of actual VOLSERS

## Automated space management

The main objective of the MGEXTSZ parameter is an attempt to prevent VSAM maximum extent limit errors for DB2-managed page sets, as follows:

- ▶ VSAM can reach maximum data set size before running out of extents. Be aware of heavily fragmented volumes, which can impede this feature. This is less of an issue starting with z/OS 1.7 where an LDS can have up to 7,257 extents. The sliding secondary rule can exceed 255 extents when 'Extent Constraint Removal' is turned on starting z/OS 1.7.
- ▶ Calculation for next extent is based on total space and not total extents because of such factors as extent consolidation.

DB2 V8 added a new function called automated space management or sliding secondary space allocation, which can be turned on automatically in DB2 V8 NFM and beyond, and is used for DB2 managed data sets (that is STOGROUP defined).

This enhancement helps to reduce the number of out-of-space conditions, eliminate need for PRIQTY and SECQTY specification on SQL CREATE, improve user productivity, and avoids the performance penalty associated with small extent sizes.

This patented solution benefits all users of DB2. It delivers autonomic selection of data set extent sizes with a goal of preventing out-of-extent errors before reaching maximum data set size. It particularly benefits users of ERP/CRM vendor applications, which have many small data sets that can grow rapidly.

The DB2 V8-introduced DSNZPARM parameter OPTIMIZE EXTENT SIZING (with a global scope), is option 6 on install panel DSNTIP7 with DB2 9. It determines what happens if a DB2 LDS is out of space and DB2 needs to create a new data set (piece). The values are YES and NO. The default value for MGEXTSZ is NO.

Sliding secondary is always on, whether you specify YES or NO. The options have the following effects:

- ▶ NO

When specifying a value for PRIQTY and SECQTY, the new data set (created after A001) will have the same allocation as A001 for primary and secondary and therefore take on the characteristics of A001's PRIQTY and SECQTY. If PRIQTY and SECQTY are not specified, the allocation works as described in the next bullet, YES.

- ▶ YES

When allocating a new data set for a new piece (after A001), the primary and secondary allocation will be the last size calculated by the sliding secondary of the previous piece. The maximum allocation does not exceed 127 cylinders for objects 16 GB or below, and 559 cylinders for objects 32 GB or 64 GB.

For example, if A001 reaches the 2 GB limit, DB2 will have to create the A002 data set. If A001's last allocation, based on a sliding scale, is 79 cylinders, A002 will be created with a primary and secondary allocation of 79 cylinders.

The maximum allocation will not exceed 127 cylinders for objects 16 GB or below, and 559 cylinders for objects 32 or 64 GB.

Sliding space management uses cylinder allocation. The default PRIQTY is as follows:

- ▶ For non-LOB table spaces and indexes: 1 cylinder
- ▶ For LOB table spaces: 10 cylinders

Sliding secondary space management can be useful and it can be applied in the following situations:

- ▶ New page sets: You are not required to specify any value for either PRIQTY or SECQTY.
- ▶ Existing page sets: Execute SQL to ALTER PRIQTY/SECQTY values to minus one (-1) plus schedule a REORG.
- ▶ If PRIQTY is specified by the user, the PRIQTY value will be honored. Otherwise, if not, the new default value as determined by either TSQTY, TSQTY\*10, or IXQTY is applied: one cylinder for non-LOB table spaces and indexes, and ten cylinders for LOB table spaces.
- ▶ If no SECQTY is specified by the user, the actual secondary quantity allocation is determined by the maximum of 10% of PRIQTY, and the minimum of calculated secondary allocation quantity size using the slide scale methodology and 559 (or 127) cylinders depending on maximum DB2 data set size. When a page set spills onto a secondary data set, the actual secondary allocation quantity will be determined and applied to the primary allocation. The progression will then continue. Prior to DB2 Version 8, the PRIQTY would have been used.
- ▶ If SECQTY is specified by the user as 0 (zero) to indicate *Do not extend*, it is always honored. The condition applies to DSNDB07 work files where many users set SECQTY to zero to prevent work files growing out of proportion.
- ▶ If SECQTY is specified by the user as greater than 0 (and MGEXTSZ is set to YES), the actual secondary allocation quantity will be the maximum of the minimum of calculated secondary allocation quantity size using the slide scale methodology and 559 (or 127) cylinders depending on maximum DB2 data set size, and the SECQTY value specified by the user. When a page set spills onto a secondary data set, the actual secondary allocation quantity will be determined and applied as the primary allocation. The progression will then continue. Prior to DB2 Version 8, the PRIQTY would have been used.

Figure 3-14 depicts the sliding scale mechanism for 64 GB data sets with 255 extents. It assumes an initial extent of one cylinder.

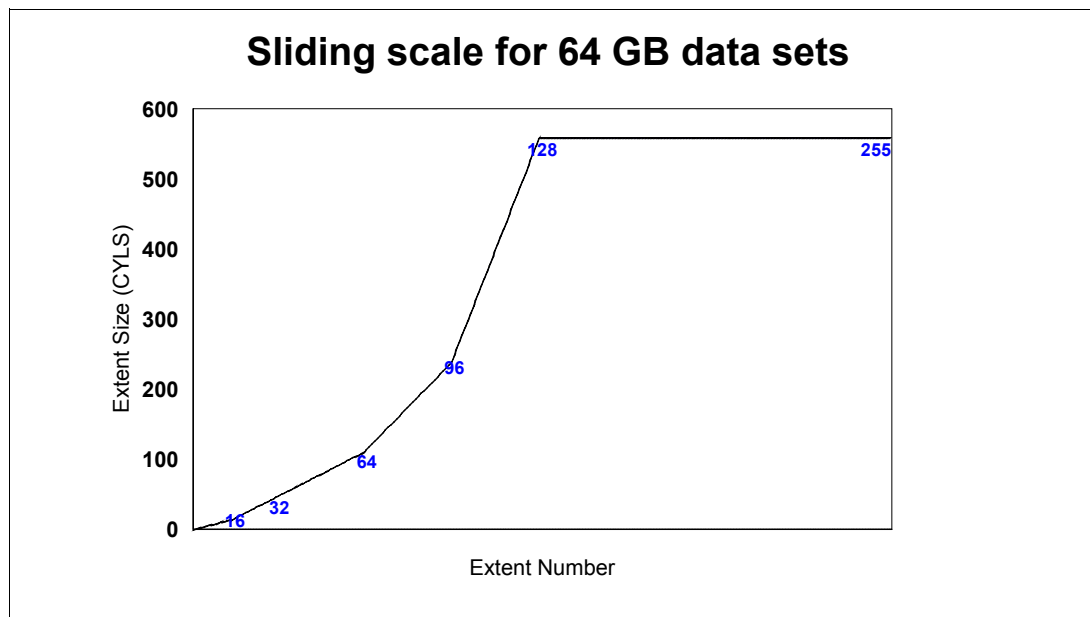


Figure 3-14 Sliding scale for 64 GB data sets



Figure 3-15 shows the sliding scale mechanism for a 16 GB data set with a maximum of 255 extents. It assumes an initial extent of one cylinder.

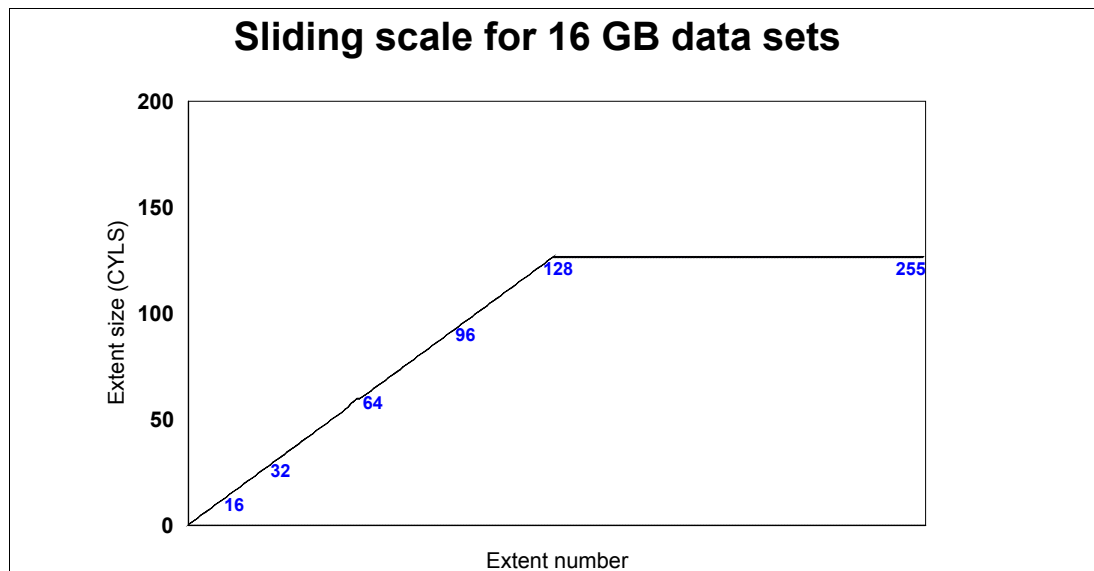


Figure 3-15 Sliding scale for 16 GB data sets

APAR PK50113 (with PTF UK36548 for DB2 V8 and UK36549 for DB2 9) introduced a new rule for assigning the primary space quantity to create the new subsequent extents of DB2-managed linear page sets. The change is meant to apply the large PRIQTY specified by the user to all of the new extents, not only the first one of the page set, so they go to the same storage group.

When a DB2-managed linear page set extends onto a new piece, the primary allocation quantity of the new piece is determined as follow:

- ▶ If DSNZPARM MGEXTSZ setting is NO
 

DB2 uses the primary quantity from the existing rule 1, as documented in *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854, under CREATE TABLESPACE statement.
- ▶ If DSNZPARM MGEXTSZ setting is YES
 

DB2 uses the calculated quantity using sliding scale algorithm, the primary quantity from the existing rule one, and the SECQTY value specified by the user are considered in setting the primary quantity.

An increasing secondary quantity size up to 127 extents is allocated and a constant number of 127 cylinders for data sets less than 1 GB or 1, 2, 4, 8 and 16 GB is used for the secondary allocation thereafter. This approach of sliding the secondary quantity minimizes the potential for wasted space by increasing the extents size slowly at first, and it avoids very large secondary allocations from extents 128-255, which most likely cause fragmentation where multiple extents have to be used to satisfy a data set extension. The solution addresses newly allocated data sets and existing data sets requiring additional extent.

Starting with z/OS 1.7, SMS-managed data sets with the Extent Constraint Removal option in the SMS data class set to YES, the theoretical extent limit has been increased to 7,257 (assuming a data set allocation over the maximum of 59 volumes, each volume with 123 extents).

DB2 can apply the sliding secondary beyond the 255 extent limit, if necessary. Requiring more than 255 extents is typically the result of heavy volume fragmentation. When discussing

extents, consider that if extent consolidation is used, in this case DB2 still provides the correct calculation based on size.

The value for PRIQTY plays a major role in the sliding allocations. The actual value applied by DB2 for default PRIQTY is determined by the applicable TSQTY or IXQTY DSNZPARMs. DSNZPARMs TSQTY and IXQTY now have global scope.

TSQTY applies to non-LOB table spaces. For LOB table spaces, a 10-times multiplier is applied to TSQTY to provide the default value for PRIQTY. IXQTY applies to indexes. DSNZPARMs TSQTY and IXQTY continue to have a default value of 0 (zero), but this value indicates that a new default value of 720 KB (1 cylinder) is to be applied. If TSQTY is set to 0, then one cylinder is the default PRIQTY space allocation for non-LOB table spaces and 10 cylinders is the default PRIQTY space allocation for LOB table spaces. If IXQTY is set to 0, then one cylinder is the default PRIQTY space allocation for indexes.

DB2 applies the following rules when calculating the allocations:

- ▶ If PRIQTY is specified by the user, the PRIQTY value will be honored. Otherwise, if not, the new default value as determined by either TSQTY,  $TSQTY \times 10$ , or IXQTY is applied: one cylinder for non-LOB table spaces and indexes, and 10 cylinders for LOB table spaces.
- ▶ If no SECQTY is specified by the user, the actual secondary quantity allocation will be determined by the maximum of 10% of PRIQTY, and the minimum of calculated secondary allocation quantity size using the slide scale methodology and 559 (or 127) cylinders depending on maximum DB2 data set size. When a page set spills onto a secondary data set, the actual secondary allocation quantity will be determined and applied to the primary allocation. The progression will then continue. Prior to DB2 Version 8, the PRIQTY would have been used.
- ▶ If SECQTY is specified by the user as 0 to indicate *Do not extend*, it is always be honored. The condition applies to DSNDB07 work files where many users set SECQTY to zero to prevent work files from growing out of proportion.
- ▶ If SECQTY is specified by the user as greater than 0 (and MGEXTSZ is set to YES), the actual secondary allocation quantity is the maximum of the minimum of calculated secondary allocation quantity size using the slide scale methodology and 559 (or 127) cylinders depending on maximum DB2 data set size, and the SECQTY value specified by the user. When a page set spills onto a secondary data set, the actual secondary allocation quantity will be determined and applied as the primary allocation. The progression then continues.

You can provide override values for TSQTY and IXQTY DSNZPARMs to avoid wasting excessive disk space. For example, on a development subsystem, TSQTY and IXQTY may be set to 48 KB for track allocation. The use of the default for PRIQTY is recorded in the associated PQTY column as minus one (-1) in the SYSTABLEPART or SYSINDEXPART catalog table.

DB2 always honors the PRIQTY value specified for the first piece (A001) by the user and recorded in the associated PQTY column in the SYSTABLEPART or SYSINDEXPART catalog table.

The next two figures show examples that summarize the data set allocations for a segmented table space as follows:

- ▶ In the first case (Figure 3-16 on page 181), TSQTY and IXQTY DSNZPARMs are set, or have been left as the default, to zero. Note the following information:
  - When MGEXTSZ is set to NO (default) and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for both primary and secondary allocation.

DB2 honors the request in your CREATE statement; system allocation and sliding allocation are not active.

- When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for primary allocation, and it becomes 15 tracks for secondary allocation.

DB2 activates the default sliding secondary allocation.

- When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, and also set DSVCI to YES, in the case of 32 KB pages, DB2 increases the allocation to a primary of two tracks and a secondary of 16.

This approach allows the 16 KB block to span the track.

- For both MGEXTSZ YES or NO, with PRIQTY 48 KB and no SECQTY, the primary allocation is one track, the secondary allocation is 15 tracks.
- For both MGEXTSZ YES or NO, with no PRIQTY and SECQTY 48 KB, both primary and secondary allocations are one cylinder.
- For both MGEXTSZ YES or NO, with no PRIQTY and no SECQTY, both primary and secondary allocations are one cylinder.
- For both MGEXTSZ YES or NO, with PRIQTY 72000 KB and no SECQTY, the primary allocation is 100 cylinders, the secondary allocation is 10 cylinders.
- For both MGEXTSZ YES or NO, with no PRIQTY and SECQTY of 72000 KB, the primary allocation is one cylinder, the secondary allocation is 100 cylinders.
- For both MGEXTSZ YES or NO, with PRIQTY 72000 and SECQTY of 0, the primary allocation is 100 cylinders, the secondary allocation is 0.

### Sample Allocations for A001 Segmented (assuming TSQTY=0 and IXQTY=0)

#### With MGEXTSZ=NO

```
PRIQTY      48
SECQTY      48

1st extent tracks . : 1
Secondary tracks . : 1
```

#### With MGEXTSZ=YES

```
PRIQTY      48
SECQTY      48

1st extent tracks . : 1
Secondary tracks . : 15
```

With MGEXTSZ=YES and DSVCI=YES. For 32K pages only - we add 2.2% to the allocation:

```
PRIQTY      48
SECQTY      48

1st extent tracks . : 2
Secondary tracks . : 16
```

#### With MGEXTSZ=NO or YES

```
PRIQTY      48
no secondary

1st extent tracks . : 1
Secondary tracks . : 15
```

```
no primary
SECQTY      48

1st extent cylinders: 1
Secondary cylinders : 1
```

```
no primary and no secondary

1st extent cylinders: 1
Secondary cylinders : 1
```

```
PRIQTY      72000
no secondary

1st extent cylinders: 100
Secondary cylinders : 10
```

#### With MGEXTSZ=NO

```
no primary
SECQTY      72000

1st extent cylinders: 1
Secondary cylinders : 100
```

```
PRIQTY      72000
SECQTY      0

1st extent cylinders: 100
Secondary cylinders : 0
```

Figure 3-16 Allocations with no system settings

- In the second case (Figure 3-17 on page 183), TSQTY and IXQTY DSNZPARMs are set to 3600 KB, equivalent to five cylinders. Note the following information:
  - When MGEXTSZ is set to NO (default) and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for both primary and secondary allocation.  
DB2 honors the request in your CREATE statement; system allocation and sliding allocation are not active.
  - When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for primary allocation, and it becomes 15 tracks for secondary allocation.  
DB2 activates the default sliding secondary allocation.
  - When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, and also set DSVCI to YES, in the case of 32 KB pages DB2 increases the allocation to a primary of two tracks and a secondary of 16.  
This enables the 16 KB block to span the track.
  - For both MGEXTSZ YES or NO, with PRIQTY 48 KB and no SECQTY, the primary allocation is one track, the secondary allocation is 15 tracks.
  - For both MGEXTSZ YES or NO, with no PRIQTY and no SECQTY, the primary allocation is five cylinders, the secondary allocation is one cylinder.
  - For both MGEXTSZ YES or NO, with PRIQTY 72000 KB and no SECQTY, the primary allocation is 100 cylinders, the secondary is 10 cylinders.
  - For both MGEXTSZ YES or NO, with no PRIQTY and SECQTY 72000 KB, the primary allocation is five cylinders, the secondary is 100 cylinders.
  - For MGEXTSZ NO, with no primary and SECQTY 48 KB, the primary allocation is 75 tracks (five cylinders) and the secondary is one track. CA is now one track.  
Notice that with APAR PK05644, instead of relying on the allocation unit, DB2 will preformat the space up to two cylinders when the page set is initialized or extended
  - For MGEXTSZ YES, with no primary and SECQTY 48 KB, the primary allocation is five cylinders and the secondary is one cylinder. CA is now 15 tracks.

## Sample Allocations for A001 Segmented (assuming TSQTY=3600 and IXQTY=3600)

<p><b>With MGEXTSZ=NO</b></p> <p>PRIQTY            48 SECQTY            48</p> <p>1st extent tracks . : 1 Secondary tracks . : 1</p> <p><b>With MGEXTSZ=YES</b></p> <p>PRIQTY            48 SECQTY            48</p> <p>1st extent tracks . : 1 Secondary tracks . : 15</p> <p><b>With MGEXTSZ=YES and DSVCI=YES. For 32K pages only - we add 2.2% to the allocation:</b></p> <p>PRIQTY            48 SECQTY            48</p> <p>1st extent tracks . : 2 Secondary tracks . : 16</p>	<p><b>With MGEXTSZ=NO or YES</b></p> <p>PRIQTY            48 no secondary</p> <p>1st extent tracks . : 1 Secondary tracks . : 15</p> <p>no primary and no secondary</p> <p>1st extent cylinders: 5 Secondary cylinders : 1</p> <p>PRIQTY            72000 no secondary</p> <p>1st extent cylinders: 100 Secondary cylinders : 10</p> <p>no primary SECQTY            72000</p> <p>1st extent cylinders: 5 Secondary cylinders : 100</p>	<p><b>With MGEXTSZ=NO</b></p> <p>no primary SECQTY            48</p> <p>1st extent tracks . : 75 Secondary tracks . : 1 TRACKS/CA-----1</p> <p><b>See PK05644 – preformat up to 2 cylinders even though allocation is in tracks. .</b></p> <p><b>With MGEXTSZ=YES</b></p> <p>no primary SECQTY            48</p> <p>1st extent cylinders: 5 Secondary cylinders : 1 TRACKS/CA-----15</p>
---	---	--

Figure 3-17 Allocations with system settings

For a striped data set, the proposed extent size is divided by the number of stripes and each stripe can have up to 255 extents. Starting with z/OS 1.7, the 255 extent rule is lifted.

When extent consolidation is in effect, started in z/OS V1.5, the secondary space allocation quantity can be smaller than specified or calculated by the sliding scale based on the physical extent number. See Table 3-8.

PTF UQ89517 for APAR PQ88665 corrects this problem by using the logical extent number together with the sliding scale.

Table 3-8 Maximum allocation for sliding secondary extents - without volume fragmentation

Max DS size in GB	Maximum allocation in cylinders	Extents to reach full size
1	127	54
2	127	75
4	127	107
8	127	154
16	127	246
32	559	172
64	559	225

In theory, starting with z/OS 1.7, VSAM LDSs are allowed to grow to 7,257 extents. Your storage administrator must set Extent Constraint Removal to YES to be able to exceed the previous limit of 255 extents per component. Be careful of the number of extents, especially for 16 GB and 64 GB data sets, because they are near the maximum allowable extents for

non-SMS managed LDSs, data sets allocated prior to z/OS 1.7, or data sets allocated with z/OS 1.7 where your Storage Administrator has set Extent Constraint Removal to NO.

Sometimes a Storage Administrator might allocate a special set of volumes in a storage group just for large data sets. Doing so typically enables large data sets to acquire larger chunks of free space while avoiding the volume fragmentation introduced by smaller data sets. Allocations to a volume in this special large storage group can be done through the storage group ACS routines using data set names of known large data sets or by a combination of size and data set pattern.

For example, you can set up the storage group ACS routines to route only data sets above 400 MB of data and if the high-level qualifier is DB2PROD and the second qualifier is DSNDBD to a special large storage group. If your storage administrator is using this space method, beware that using the default PRIQTY may no longer work because for non-LOB data sets the allocation is one cylinder. In this case, you must be explicit about the data set name or add a PRIQTY.

### Determining the size for secondary allocation

Sliding secondary allocation and several other factors can influence the size of the secondary allocation. In case of a failure you might find a mismatch between the value reported in the VTOC and the real allocation.

In a sample scenario, the table space was defined as SMS managed, with sliding secondary allocation activated, and allocated with PRIQTY 720, SECQTY 720 (where the quantity is in kilobytes).

After extending a data set with INSERTs, Example 3-31 shows the output of a list catalog.

**Example 3-31** LISTCAT output for checking allocations

1	IDCAMS	SYSTEM SERVICES	TIME: 15:25:27	06/30/10	PAGE	1
0						
		LISTC ENT(DB9AU.DSNDBC.DSNDB04.JOHNEXT.I0001.A001) ALL	01240001			
0		OCUSTER ----- DB9AU.DSNDBC.DSNDB04.JOHNEXT.I0001.A001				
		IN-CAT --- UCAT.DB9A				
		HISTORY				
		DATASET-OWNER----PAOLOR9	CREATION-----2010.181			
		RELEASE-----2	EXPIRATION-----0000.000			
		SMSDATA				
		STORAGECLASS -----DB9A	MANAGEMENTCLASS---MCDB22			
		DATACLASS -----(NULL)	LBACKUP ---0000.000.0000			
		EATTR----- (NULL)				
		BWO STATUS-----00000000	BWO TIMESTAMP---000000 00:00:00.0			
		BWO----- (NULL)				
		RLSDATA				
		LOG ----- (NULL)	RECOVERY REQUIRED --(NO)	FRLOG ----- (NULL)		
		VSAM QUIESCED ----- (NO)	RLS IN USE ----- (NO)			
0		LOGSTREAMID----- (NULL)				
		RECOVERY TIMESTAMP LOCAL-----X'0000000000000000'				
		RECOVERY TIMESTAMP GMT-----X'0000000000000000'				
		PROTECTION-PSWD----- (NULL)	RACF----- (NO)			
		ASSOCIATIONS				
		DATA-----DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001				
0		DATA ----- DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001				
		IN-CAT --- UCAT.DB9A				
		HISTORY				
		DATASET-OWNER----PAOLOR9	CREATION-----2010.181			
		RELEASE-----2	EXPIRATION-----0000.000			
		ACCOUNT-INFO----- (NULL)				
		PROTECTION-PSWD----- (NULL)	RACF----- (NO)			
		ASSOCIATIONS				
		CLUSTER--DB9AU.DSNDBC.DSNDB04.JOHNEXT.I0001.A001				

```

ATTRIBUTES
KEYLEN-----0          AVGLRECL-----0          BUFSPACE-----8192          CISIZE-----4096
RKP-----0          MAXLRECL-----0          EXCPEXIT----- (NULL)          CI/CA-----180
SHROPTNS(3,3)    SPEED    UNIQUE          NOERASE    LINEAR          NOWRITECHK    NOIMBED          NOREPLICAT
UNORDERED          REUSE    NONSPANNED

STATISTICS
REC-TOTAL-----0          SPLITS-CI-----0          EXCPS-----0
REC-DELETED-----0          SPLITS-CA-----0          EXTENTS-----1
REC-INSERTED-----0          FREESPACE-%CI-----0          SYSTEM-TIMESTAMP:
REC-UPDATED-----0          FREESPACE-%CA-----0          X'0000000000000000'
REC-RETRIEVED-----0          FREESPC-----0

ALLOCATION
SPACE-TYPE-----CYLINDER    HI-A-RBA-----15482880
SPACE-PRI-----1          HI-U-RBA-----15482880
SPACE-SEC-----1

VOLUME
VOLSER-----SBOX1X          PHYREC-SIZE-----4096          HI-A-RBA-----15482880          EXTENT-NUMBER-----1
1IDCAMS  SYSTEM SERVICES          TIME: 15:25:27          06/30/10          PAGE          2
0          DEVTYPE-----X'3010200F'    PHYRECS/TRK-----12          HI-U-RBA-----15482880          EXTENT-TYPE-----X'40'
          VOLFLAG-----PRIME          TRACKS/CA-----15
          EXTENTS:
          LOW-CCHH-----X'00350000'    LOW-RBA-----0          TRACKS-----315
          HIGH-CCHH-----X'0049000E'    HIGH-RBA-----15482879

1IDCAMS  SYSTEM SERVICES          TIME: 15:25:27          06/30/10          PAGE          3
0          THE NUMBER OF ENTRIES PROCESSED WAS:
          AIX -----0
          ALIAS -----0
          CLUSTER -----1
          DATA -----1
          GDG -----0
          INDEX -----0
          NONVSAM -----0
          PAGESPACE -----0
          PATH -----0
          SPACE -----0
          USERCATALOG -----0
          TAPELIBRARY -----0
          TAPEVOLUME -----0
          TOTAL -----2
0          THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
OIDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0
OIDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0

```

The output shows the allocation CYL(1,1), primary and secondary of one cylinder, and only one extent, for the 4 KB page data set DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001. As for all non-LOB sliding secondary objects, settings are PRIQTY=180, SECQTY=180. We can determine what exactly happened to the data set by activating the DB2 performance trace for IFCID 258 - Data Set Extend Activity. Record 258 is written every time a data set is extended.

Example 3-32 lists the IBM OMEGON XE for DB2 Performance Expert (OMEGAMON® PE) output for the collected 258 records.

#### Example 3-32 Extend trace output

```

1                                     OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V4R2)          PAGE: 1
                                     EXECUTION LOG                                     RUN DATE: 06/30/10 15:21:59.39

MSG.ID.    DESCRIPTION
-----
FPEC2001I  COMMAND INPUT FROM DDNAME SYSIN
0          DB2PM GLOBAL(TIMEZONE(+7:00))
0          INCLUDE (IFCID(258)))
0          RECTRACE TRACE LEVEL(LONG)
0          EXEC

```

FPEC1999I SYSTEM INITIALIZATION COMPLETE. RETURN CODE 0

OFPEC0999I EXECUTION COMPLETE. RETURN CODE 0

1 LOCATION: DB9A OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V4R2)  
GROUP: N/P RECORD TRACE - LONG

PAGE: 1-1

REQUESTED FROM: NOT SPECIFIED

TO: NOT SPECIFIED

ACTUAL FROM: 06/30/10 12:21:10.73

PAGE DATE: 06/30/10

MEMBER: N/P

SUBSYSTEM: DB9A

DB2 VERSION: V9

OPRMAUTH

CONNECT

INSTANCE

END\_USER

WS\_NAME

TRANSACT

ORIGAUTH

CORRNAME

CONNTYPE

RECORD TIME

DESTNO ACE

IFC

DESCRIPTION

DATA

PLANNAME

CORRNMBR

TCB CPU TIME

ID

N/P

N/P

N/P

C63548A60B27

'BLANK'

N/P

N/P

12:21:10.73651447

N/P

103

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:10.736508

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

360

HIGH ALLOC BEFORE

:

180

HIGH ALLOC AFTER

:

540

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

N/P

N/P

N/P

C63548A9F329

'BLANK'

N/P

N/P

12:21:14.84368209

N/P

106

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:14.843673

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

540

HIGH ALLOC BEFORE

:

540

HIGH ALLOC AFTER

:

1080

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

N/P

N/P

N/P

C63548AD1062

'BLANK'

N/P

N/P

12:21:18.27211734

N/P

115

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:18.272110

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

720

HIGH ALLOC BEFORE

:

1080

HIGH ALLOC AFTER

:

1800

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

N/P

N/P

N/P

C63548B060A1

'BLANK'

N/P

N/P

12:21:21.84634406

N/P

118

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:21.846336

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

900

HIGH ALLOC BEFORE

:

1800

HIGH ALLOC AFTER

:

2700

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

OPRMAUTH

CONNECT

INSTANCE

END\_USER

WS\_NAME

TRANSACT

ORIGAUTH

CORRNAME

CONNTYPE

RECORD TIME

DESTNO ACE

IFC

DESCRIPTION

DATA

PLANNAME

CORRNMBR

TCB CPU TIME

ID

N/P

N/P

N/P

C63548A60B27

'BLANK'

N/P

N/P

12:21:10.73651447

N/P

103

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:10.736508

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

360

HIGH ALLOC BEFORE

:

180

HIGH ALLOC AFTER

:

540

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

N/P

N/P

N/P

C63548A9F329

'BLANK'

N/P

N/P

12:21:14.84368209

N/P

106

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:14.843673

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

540

HIGH ALLOC BEFORE

:

540

HIGH ALLOC AFTER

:

1080

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

N/P

N/P

N/P

C63548AD1062

'BLANK'

N/P

N/P

12:21:18.27211734

N/P

115

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:18.272110

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

720

HIGH ALLOC BEFORE

:

1080

HIGH ALLOC AFTER

:

1800

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

N/P

N/P

N/P

C63548B060A1

'BLANK'

N/P

N/P

12:21:21.84634406

N/P

118

1

258

DATA SET EXT.

ACTIVITIES

N/P

NETWORKID:

USIBMSC

LUNAME:

SCPDB9A

LWSEQ:

1

DATA SET NAME

:

DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001

TIMESTAMP

:

06/30/10 19:21:21.846336

DATABASE NAME

:

DSNDB04

DBID

:

4

TABLESPACE NAME

:

JOHNEXT

PSID

:

562

PRIMARY QUANTITY

:

180

SEC. QUANTITY

:

900

HIGH ALLOC BEFORE

:

1800

HIGH ALLOC AFTER

:

2700

MAX DS SIZE

:

524288

EXTENTS BEFORE

:

1

EXTENTS AFTER

:

1

MAX EXTENTS

:

251

VOLUMES BEFORE

:

1

VOLUMES AFTER

:

1

MAX VOLUMES

:

59

1 LOCATION: DB9A OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V4R2)  
GROUP: N/P RECORD TRACE - LONG

PAGE: 1-2

REQUESTED FROM: NOT SPECIFIED

TO: NOT SPECIFIED

ACTUAL FROM: 06/30/10 12:21:10.73

PAGE DATE: 06/30/10

MEMBER: N/P

SUBSYSTEM: DB9A

DB2 VERSION: V9

OPRMAUTH	CONNECT	INSTANCE	END_USER	WS_NAME					TRANSACT				
ORIGAUTH	CORRNAME	CONNTYPE	RECORD TIME	DESTNO	ACE	IFC	DESCRIPTION	DATA					
PLANNAME	CORRNMBR		TCB CPU TIME			ID							
N/P	N/P	C63548B060A1	N/P	N/P				N/P					
N/P	N/P	'BLANK'	12:21:22.33592926	120	1	258	DATA SET EXT.	NETWORKID:	USIBMSC	LUNAME:	SCPDB9A	LWSEQ:	1
N/P	N/P		N/P				ACTIVITIES						
-----													
DATA SET NAME : DB9AU.DSNDBD.DSNDB04.JOHNEXT.I0001.A001													
DATABASE NAME : DSNDB04 DBID : 4													
TABLESPACE NAME : JOHNEXT PSID : 562													
PRIMARY QUANTITY : 180 SEC. QUANTITY : 1080													
HIGH ALLOC BEFORE : 2700 HIGH ALLOC AFTER : 3780													
EXTENTS BEFORE : 1 EXTENTS AFTER : 1													
VOLUMES BEFORE : 1 VOLUMES AFTER : 1													
-----													
TIMESTAMP : 06/30/10 19:21:22.335921													

1 LOCATION: DB9A OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V4R2)  
GROUP: N/P RECORD TRACE - LONG

PAGE: 1-3

REQUESTED FROM: NOT SPECIFIED

TO: NOT SPECIFIED

MEMBER: N/P



SUBSYSTEM: DB9A  
DB2 VERSION: V9  
0

ACE NUMBER	ACE ADDRESS	ACE NUMBER	ACE ADDRESS	ACE NUMBER	ACE ADDRESS	ACE NUMBER	ACE ADDRESS	ACE NUMBER	ACE ADDRESS
1	X'21D65618'								

ORECORD TRACE COMPLETE

ACTUAL FROM: 06/30/10 12:21:10.73

In the trace, the allocations are shown in pages, as you would see in the DB2 catalog. There are five secondary allocations with sliding sizes of 360, 540, 720, 900, and 1080 4-KB CIs/pages in addition to the primary allocation of 180, for a total of 3780 4-KB, corresponding to the high RBA.

In this case, there is still only one extent because of VSAM *extent consolidation*<sup>4</sup>. The system consolidates adjacent extents for VSAM data sets when extending on the same volume. VSAM extent consolidation is automatic and requires no action on your part. If the extents are adjacent, the new extent is incorporated into the previous extent. In this case, LISTC indicates a logical extent 2 of 1 cylinder; it is really two cylinders.

Besides DB2 sliding allocation and VSAM extent consolidation, there are other possible reasons for secondary extents discrepancies:

- ▶ Part of the data set is allocated on an EAV in the cylinder managed area
- ▶ DEFRAG with CONSOLIDATE keyword was run
- ▶ DFSMSdss with CONSOLIDATE function was run
- ▶ DB2 ALTER increased the space but REORG or LOAD REPLACE have not been run yet.

### 3.13.8 Utility work data sets

A number of considerations for the utility work data sets are described in this section:

- ▶ Define automatic class selection (ACS) routines for data set allocation
- ▶ Stripe SYSREC, SYSUT1, and SORTOUT
- ▶ Ensure correct SORTWORK data set placement
- ▶ Prevent OEM override of space allocation
- ▶ Ensure sufficient free space in sort work pool

#### Define ACS routines for data set allocation

The ACS routines can use different attributes in a job as variables. ACS routines can examine many SMS read-only variables such as job name, RACF® group, LPAR name, Sysplex name, user name, job/step name and accounting information. These SMS read-only variables are listed in the “DFSMSrmm SMS ACS Support” Technote:

<http://www.redbooks.ibm.com/abstracts/tips0530.html?Open>

Use ACS routines to define policies for enforcing the data set naming standards and to control data set placement, device type, volume selection, and space usage for both DASD and TAPE data sets for SYSUT1, SORTOUT and other data sets. Set up ACS routines to exclude the following data sets from VIO:

- ▶ SORTWK\*
- ▶ STATWK\*
- ▶ DATAWK\*
- ▶ DAnnWK\*
- ▶ STnnWK\*
- ▶ WnnWK\*

<sup>4</sup> For details, see z/OS V1R11.0 DFSMS Using Data Sets, SC26-7410

## Stripe SYSREC, SYSUT1, and SORTOUT

The use of DFSMS striping can provide a significant reduction in elapsed time for certain utility processing. See Table 3-9 for which DDNAMEs can benefit from DFSMS striping when you are dealing with larger objects and certain utilities.

Table 3-9 Utilities that can benefit DFSMS striping

DB2 utility	DDNAMEs for striping
LOAD	SYSREC, SYSUT1 and SORTOUT
REORG INDEX	SYSUT1
REORG TABLESPACE	SYSREC, SYSUT1 and SORTOUT

## Ensure correct SORTWORK data set placement

Ensure that sort-work allocations are directed to those devices with the fastest available random I/O service times. Spread I/O for sort work files by assigning logical volumes on different physical devices to avoid cache and channel contention. In general, sort-work EXCPs are large, which might have a noticeable impact on measured disconnect times for other applications. Therefore, isolate sort work devices particularly from data with critical response time requirements

## Ensure sufficient free space in sort work pool

As a general rule, the work pool should average over 20% free space to allow for optimal allocation of sort-work data sets and reduce the risk of failures because of lack of work space (messages ICE083A, ICE046A, and so on). If the percent free space of the work pool is low, increase the number of volumes in the work pool or reduce the number of concurrently running utilities.

## Tape data sets

DB2 9 supports 256 KB block sizes for tapes. To enable the use of 256 KB block sizes, a change needs to be made in the DEVSUPxx member of SYS1.PARMLIB; set TAPEBLKSZLIM to 262144. This setting can also be dynamically activated in SDSF by entering the following command:

```
T DEVSUP=xx
```

The ideal, if available, is to use a high capacity drive for best tape performance. If using a single tape drive, choose the B-side, which gives approximately 7% higher throughput than the A-side.

## 3.14 Format and preformat of DB2 data sets

DB2 data sets are not ready for write activity when they are first created. A preformat operation takes place on the data set to make it usable for inserts or loads.

DB2 9 preformats part of the a data set up to 16 cylinders at a time. Preformat falls into two categories, synchronous and asynchronous. Synchronous preformat is performed for example when a new extent is created. Asynchronous preformat is performed for most other operations. Asynchronous preformat does not trigger the format writes when the end of its already formatted space is reached, rather earlier, based on a DB2 decided threshold.

DB2 9 preformats part of the a data set up to 16 cylinders at a time; DB2 V8 preformats part of the data set up to two cylinders or two tracks depending on maintenance level, and the

overall size. Prior to DB2 9, high volume inserts at times suffered from bottlenecks caused by smaller preformat quantities. the probability of this bottleneck with DB2 9 is greatly diminished because up to 16 cylinders are preformatted at one time.

Certain utilities use format writes instead of preformat. Because the utility knows how much data will be written, instead of preformatting a page and writing data, which now performs two operations, certain utilities perform one operation with a format-write and writing of the data.

DB2 preformats up to 16 cylinders at a time. If the allocation is for five tracks, all five tracks are preformatted. If the allocation is 16 cylinders, all 16 cylinders are preformatted. If the allocation is 17 cylinders, only the first 16 cylinders are preformatted. If 1,000 cylinders are allocated only the first 16 cylinders are preformatted.

IDCAMS LISTCAT can be executed to better understand how the allocations work. *High allocated RBA* refers to the amount of disk space allocated; *high used RBA* refers to the amount of disk that has been preformatted/formatted ready to be used by DB2 or has already been used.

To understand what LISTCAT is displaying, we review DB2 and disk basics:

- ▶ 15 tracks = 1 cylinder
- ▶ Each track has 56,664 bytes, but DB2 uses only 49,152 bytes (48 KB).
- ▶ DDL is generally created in terms of kilobytes, therefore, if one track is requested, the allocation is 48:
 

$48 * 1024 = 1\text{KB} = 49,152 \text{ bytes}$   
 One cylinder is 720:  
 $48\text{KB} * 15 \text{ tracks} = 720$
- ▶ For DB2 LDSs, LISTCAT displays information based on the 48 KB per track, not 56,664 bytes per track.

Returning to the example of 1,000 cylinders, after the DDL creating the data set is executed, the output is as follows:

```
ALLOCATION
SPACE-TYPE-----CYLINDER      HI-A-RBA-----737280000
SPACE-PRI-----1000          HI-U-RBA-----11796480
SPACE-SEC-----100
```

**HI-A and HI-U:** *High allocated* (HI-A) refers to the total amount of space allocated; *high used* (HI-U) refers to the amount of space already used and preformatted.

The high allocated is 737280000 bytes. We get this by multiplying 720 by 1024 by 1000. The value 720 refers to the number of bytes per cylinder used by DB2 multiplied by the next number 1024, 1024 stands for 1 KB (1024 bytes), multiplied by 1,000 which is the total number of cylinders:

$$720 * 1024 * 1000 = 737280000$$

The calculation based on tracks instead of cylinders is as follows:

$$48 * 15 * 1024 * 1000 = 737280000$$

The value of 48 is the number of bytes occupying a track multiplied later by 1024; 15 is the number of tracks per cylinder; 1024 is 1 KB; 1000 is 1,000 cylinders. Therefore 737280000 translates to 1,000 cylinders worth of data that DB2 has physically allocated.

The HI-U is 11796480 bytes, same formula as the previous formula:

$720 * 1024 * 16 = 11796480$

The 720 refers to the number of bytes per cylinder used by DB2, 1024 stands for 1KB (1024 bytes), multiplied by 16 cylinders. Therefore 11796480 translates to 16 cylinders worth of data that DB2 has formatted or preformatted and is ready to use.

The discrepancy between high allocated and high used indicates that not all 1,000 cylinders allocated can be used by DB2, only 16 cylinders worth at this time until more preformatting is executed. The next preformat will be for another 16 cylinders, in which case high used is increased by another 11796480 bytes. This increase is done until the high allocated 737280000 bytes have been exhausted and a new extent is required.

The LOAD and REORG utilities provide a PREFORMAT option, which allows DB2 to preformat the entire data set. In this case, the high used and high allocated values are the same. The disadvantage is the extra time during the utility execution. The advantage is that DB2 does not have to preformat data during heavy insert activity. Allocating the next extent starts the normal preformat over again. See Figure 3-18.

```
CREATE TABLESPACE JOHNITS1
  USING STOGROUP SYSDEFLT
  PRIQTY          72000      100 cyl (48k per track * 15 tracks per cyl)=720*100
  SECQTY          3600      5 cyl (48k per track * 15 tracks per cyl)=720*5
  LOCKSIZE        ANY      (keep in mind PRIQTY and SECQTY are specified in KB)
  CLOSE           NO
  BUFFERPOOL      BP2
  FREEPAGE        0
  PCTFREE         0;

LISTCAT output:
ALLOCATION
SPACE-TYPE-----CYLINDER    HI-A-RBA-----3686400    5 cylinders=3600*1024(1kb)
SPACE-PRI-----100          HI-U-RBA-----11796480    preformat 16 cylinders in kb
SPACE-SEC-----5

CREATE TABLESPACE JOHNITS1
  USING STOGROUP SYSDEFLT
  PRIQTY          3600      5 cyl (48k per track * 15 tracks per cyl)=720*5
  SECQTY          720      1 cyl (48k per track * 15 tracks per cyl)=720*1
  LOCKSIZE        ANY      (keep in mind PRIQTY and SECQTY are specified in KB)
  CLOSE           NO
  BUFFERPOOL      BP2
  FREEPAGE        0
  PCTFREE         0;

LISTCAT output:
ALLOCATION
SPACE-TYPE-----CYLINDER    HI-A-RBA-----3686400    5 cylinders=3600*1024(1kb)
SPACE-PRI-----5           HI-U-RBA-----3686400    preformat 5 cylinders in kb
SPACE-SEC-----1
```

Figure 3-18 Preformat example

Some customers use DFSMSHsm or DFSMSdss to release the unused space in the LDSs in their non-production environment. The data set must be allocated with EF and non-Guaranteed Space. This can mean a significant savings in disk space, but performance suffers slightly when more data is needed because a new extent must be created. For example, you have a data set with 1,000 cylinders and only 16 cylinders are used. You run a DFSMSdss RELEASE for the data set, now only 16 cylinders are allocated.

Consider the following information and see Example 3-33:

- ▶ If the data set will not be changed and therefore not increase, releasing unused space can gain back some disk.
- ▶ If RELEASE is executed on the primary amount and has not yet allocated the secondary amount, what will the overall space request look like? For example, allocation is in cylinders, 1,000 primary, 100 secondary; 1,000 cylinders has been allocated, but only 16 cylinders are used. You run the RELEASE, the size is now 16 cylinders. Did you expect the 1,000 cylinders to be there? Any issues now with the allocation for overall space?
- ▶ There can be a slight performance degradation for the next format for the next extent.
- ▶ RELEASE has parameters to protect against various potential issues, such as using MINSECQTY and MINTRACKSUNUSED.

*Example 3-33 Extended format table space with no Guaranteed Space assigned after CREATE*

---

```

CREATE TABLESPACE JOHNRLSE
  USING STOGROUP JOHNEFEA
  PRIQTY          720000
  SECQTY          72000
  LOCKSIZE       ANY
  CLOSE          NO
  BUFFERPOOL     BPO
  FREEPAGE       0
  PCTFREE        0;

LISTCAT ENTRIES(DB9AU.DSNDBC.DSNDB04.JOHNRLSE.I0001.A001) ALL
ATTRIBUTES -EXTENDED
ALLOCATION
  SPACE-TYPE-----CYLINDER      HI-A-RBA-----737280000
  SPACE-PRI-----1000           HI-U-RBA-----11796480
  SPACE-SEC-----100

```

---

Example 3-34 shows success after space was stopped.

*Example 3-34 Successful DFSMSdss RELEASE job after the table space was stopped*

---

```
//STEP1 EXEC PGM=ADRDSSU,REGION=4M
//DASD0002 DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
RELEASE INCLUDE(DB9AU.DSNDB%.DSNDB04.JOHNRLSE.**)
```

```
- RELEASE INCLUDE(DB9AU.DSNDB%.DSNDB04.JOHNRLSE.**) 00110001
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'RELEASE '
ADR109I (R/I)-RI01 (01), 2010.132 17:31:07 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
OADR006I (001)-STEND(01), 2010.132 17:31:07 EXECUTION BEGINS
OADR458I (001)-RLSE0(01), TOTAL NUMBER OF TRACKS ON VOLUME SBOX0H IS
0 0000150255
OADR458I (001)-RLSE0(02), USED NUMBER OF TRACKS ON VOLUME SBOX0H IS
0 0000048603
OADR457I (001)-RLSE0(01), THE NUMBER OF TRACKS MADE AVAILABLE ON VOLUME SBOX0H IS
0 0000014760
OADR456I (001)-RLSE0(01), THE NUMBER OF DATA SETS PROCESSED ON VOLUME SBOX0H IS
0 000001
OADR454I (001)-RLSE0(01), THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED
0 DB9AU.DSNDBC.DSNDB04.JOHNRLSE.I0001.A001 COMPONENT
DB9AU.DSNDBD.DSNDB04.JOHNRLSE.I0001.A001
OADR006I (001)-STEND(02), 2010.132 17:31:12 EXECUTION ENDS
OADR013I (001)-CLTSK(01), 2010.132 17:31:12 TASK COMPLETED WITH RETURN CODE 0000
OADR012I (SCH)-DSSU (01), 2010.132 17:31:12 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS
0000
```

---

Example 3-35 shows successful execution

*Example 3-35 LISTCAT output after RELEASE was successfully executed*

---

```
LISTCAT ENTRIES(DB9AU.DSNDBC.DSNDB04.JOHNRLSE.I0001.A001) ALL
ATTRIBUTES -EXTENDED
ALLOCATION
SPACE-TYPE-----CYLINDER HI-A-RBA-----11796480
SPACE-PRI-----1000 HI-U-RBA-----11796480
SPACE-SEC-----100
```

---

This examples shows the following information:

- ▶ The data set is allocated with 1,000 cylinders, but only 16 cylinders used.
- ▶ STOP of the object
- ▶ DFSMSdss RELEASE job runs successfully
- ▶ START of the object
- ▶ The data set is now allocated with 16 cylinders and space used is 16 cylinders - the same value.



## System managed DB2 data

This chapter is designed to familiarize the DB2 database administrator (DBA) with system managed storage and getting to plan for DB2 data managed by System Managed Storage (SMS).

First, we present the concepts and components of SMS by discussing the 4.1.2, “DFSMS/MVS components” on page 194 and their benefits.

We then discuss the functionality of SMS by explaining the use of automatic class selection routines, SMS classes, and naming standards.

Finally we describe DB2 databases from the point of view of their attributes for SMS management, and provide examples for these databases, including examples of SMS Data, Storage, Management Classes, and the storage group for DB2 objects.

This chapter contains the following topics:

- ▶ Concepts and components
- ▶ Storage management with DFSMS
- ▶ SMS examples for DB2 databases

## 4.1 Concepts and components

The continued growth of data requires the need for a more effective and efficient way of managing both data and the storage on which it resides. SMS was introduced in 1988 both as a concept and then as a group of MVS<sup>1</sup> products to provide a solution for managing disk storage. Based upon user specifications, SMS can determine data placement, backup, migration, performance, and expiration. The goals of SMS are as follows:

- ▶ A reduction in the number of personnel that is required to manage that data, by allowing the system to manage as much as possible
- ▶ A reduction in labor-intensive related tasks of disk management, by centralizing control, automating tasks, and providing interactive tools
- ▶ A reduction in the necessity for user knowledge of placement, performance, and space management of data

### 4.1.1 Evolution

Although initially a concept, with a small number of products offering limited functionality, the introduction of DFSMS<sup>2</sup> has provided the functions needed for a comprehensive storage management subsystem, which provides the following benefits:

- ▶ Management of storage growth
- ▶ Improvement of storage utilization
- ▶ Centralized control of external storage
- ▶ Exploitation of the capabilities of available hardware
- ▶ Management of data availability

With each subsequent release of the product, more features have been introduced that further exploit the concepts of SMS managed data, and this is likely to continue, for example, advanced functions for all types of VSAM files, which require the use of the extended addressability (EA) attribute in the Data Class. It is therefore important to understand that those customers who have taken the time and effort to implement an SMS policy ultimately gain more from DFSMS enhancements than those who have not.

The characteristics of a DB2 system allows for the management of its data by SMS. However, considerations and choices need to be made to tailor it to suit the individual customer's environment. These considerations are discussed in the following sections.

### 4.1.2 DFSMS/MVS components

DFSMS/MVS provides and enhances functions formerly provided by MVS/DFP, Data Facility Data Set Services (DFDSS), and the Data Facility Hierarchical Storage Manager (DFHSM). The product is now easier to install and order than the combination of the earlier offerings. This section describes the main components of the DFSMS/MVS family:

- ▶ DFSMSdftp
- ▶ DFSMSdss
- ▶ DFSMSHsm
- ▶ DFSMSrmm
- ▶ Tivoli OMEGAMON XE for Storage

---

<sup>1</sup> The term MVS (z/OS) refers to the family of products which, when combined, provide a fully integrated operating system.

<sup>2</sup> The term DFSMS refers to the family of products which, when combined, provide a system-managed storage environment.



## DFSMSdfp

The Data Facility Product (DFP) component provides storage, data, program, tape, and device management functions.

DFP is responsible for the creation and accessing of data sets. It provides a variety of different access methods to organize, store and retrieve data, through program and utility interfaces to VSAM, partitioned, sequential, and direct access types.

DFP also provides the Interactive Storage Management Facility (ISMF) which allows the definition and maintenance of storage management policies interactively. It is designed to be used by both users and storage administrators.

### *ISMF for the user*

Figure 4-1 shows the ISMF Primary option menu panel displayed for a user. Various options allow the user to list the available SMS classes, display their attributes, and build lists of data sets based upon a selection criteria. These lists are built from VTOC or catalog information, and are tailored by using filtering, sorting, masking criteria. This panel is selected from the ISPF/PDF<sup>3</sup> primary menu (dependent upon site customizing).

```
ISMF PRIMARY OPTION MENU - z/OS DFSMS V1 R11
Enter Selection or Command ==>

Select one of the following options and press Enter:

0  ISMF Profile           - Change ISMF User Profile
1  Data Set               - Perform Functions Against Data Sets
2  Volume                 - Perform Functions Against Volumes
3  Management Class       - Specify Data Set Backup and Migration Criteria
4  Data Class             - Specify Data Set Allocation Parameters
5  Storage Class          - Specify Data Set Performance and Availability
9  Aggregate Group        - Specify Data Set Recovery Parameters
L  List                   - Perform Functions Against Saved ISMF Lists
R  Removable Media Manager - Perform Functions Against Removable Media
X  Exit                   - Terminate ISMF
```

Figure 4-1 ISMF Primary option menu for users

Some storage administrators restrict the use of ISMF outside of their area. ISMF provides a lot of information to the user. Storage administrators can use RACF to allow users to view certain information and restrict other information.

### *ISMF for the storage administrator*

Figure 4-2 on page 196 shows the ISMF Primary option menu displayed for a storage administrator. Options allow lists to be built from volume selection criteria (storage group), and the Data Class, Storage Class, and Management Class facilities, allowing an authorized individual to define, alter, copy, and delete SMS classes, volumes, and data sets. Again, these lists are built from VTOC or catalog information, and tailored in the same way. This panel is selected from the ISPF/PDF primary menu (dependent upon site customizing).

<sup>3</sup> Interactive System Productivity Facility (ISPF)/ Program Development Facility (PDF)

```

ISMF PRIMARY OPTION MENU - z/OS DFSMS V1 R11
Selection or Command ==>

0 ISMF Profile           - Specify ISMF User Profile
1 Data Set               - Perform Functions Against Data Sets
2 Volume                 - Perform Functions Against Volumes
3 Management Class       - Specify Data Set Backup and Migration Criteria
4 Data Class             - Specify Data Set Allocation Parameters
5 Storage Class          - Specify Data Set Performance and Availability
6 Storage Group          - Specify Volume Names and Free Space Thresholds
7 Automatic Class Selection - Specify ACS Routines and Test Criteria
8 Control Data Set       - Specify System Names and Default Criteria
9 Aggregate Group        - Specify Data Set Recovery Parameters
10 Library Management    - Specify Library and Drive Configurations
11 Enhanced ACS Management - Perform Enhanced Test/Configuration Management
C Data Collection        - Process Data Collection Function
G Report Generation      - Create Storage Management Reports
L List                   - Perform Functions Against Saved ISMF Lists
P Copy Pool              - Specify Pool Storage Groups for Copies
R Removable Media Manager - Perform Functions Against Removable Media
X Exit                   - Terminate ISMF

```

Figure 4-2 ISMF Primary option menu for storage administrator

## DFSMSdss

The Data Set Services (DSS) component is a disk storage management tool. It can be invoked using ISMF or by running batch jobs. This section describes DSS capabilities.

### **Functionality**

The DSS component is able to perform a variety of space management functions. Of these, the most common are as follows:

<b>COMPRESS</b>	Compresses partitioned data sets.
<b>CONSOLIDATE</b>	Consolidates multiple extent data sets on a particular volume.
<b>CONVERTV</b>	Converts existing volumes to and from SMS management without data movement.
<b>COPY</b>	Performs data set, volume, and track movement, allowing the movement of data from one disk volume to another, including unlike device types.
<b>COPYDUMP</b>	Allows the generation of 1 - 255 copies of DFSMSdss produced dump data. The data to be copied can be tape or disk-based, and copies can be written to tape or disk volumes.
<b>DEFRAG</b>	Reduces or eliminates free-space fragmentation on a disk volume.
<b>DUMP</b>	Performs the dumping of disk data to a sequential volume (either tape or disk). Processing can be of two types:  <i>Logical processing</i> , which is data-set oriented, which means it performs against data sets independently of the physical device format.  <i>Physical processing</i> , which can perform against data sets, volumes, and tracks, but moves data at the track-image level.

<b>PRINT</b>	Prints both VSAM and non-VSAM data sets, tracks ranges, or a VTOC.
<b>RELEASE</b>	Releases allocated but unused space from all eligible sequential, partitioned, and extended format VSAM data sets without the Guaranteed Space attribute enabled.
<b>RESTORE</b>	Restores data to disk volumes from DFSMSdss produced dump volumes, as individual data sets, an entire volume, or a range of tracks. Again, <i>logical</i> or <i>physical</i> processing can be used.

Most functions can be simulated to verify that the correct action is taken prior to the execution. Simulation is done by adding PARM= ' TYPRUN=NORUN ' parameter to the EXEC DD statement. When verified, PARM can be removed and the process executed.

### ***Filtering***

The DSS component uses a filtering process to select data sets based upon the following specified criteria:

- ▶ Fully or partially qualified data set name
- ▶ Last reference data
- ▶ Size of data sets
- ▶ Number of extents
- ▶ Allocation type (CYLS, TRKS, BLKS)
- ▶ SMS Class

For example, if DSNZPARMs indicates ARCRETN=9999, archive log data sets are not expired. In this scenario, we also create our archive log data sets on disk. The storage administrator is concerned because the pool is not large enough to handle an ever growing process. The decision was made to delete the archive log data sets over 15 days not referenced as shown in Example 4-1. One alternative is to associate the archive log data sets with a Management Class to expire the data sets by DFSMSHsm. Another alternative is to use DFSMSdss filtering to expire the data sets.

*Example 4-1 DFSMSdss job that deletes archive logs after 15 days, not referenced*

---

```
//STEP1   EXEC   PGM=ADRDSSU,REGION=4M
//DASD0002 DD   DUMMY
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
DUMP DATASET(INCLUDE(DB9AU.ARCHLOG%,**)) -
BY(REFDT LT *,-15)) -
OUTDDNAME(DASD0002) DELETE PURGE
```

---

### ***Converting data to SMS***

Data sets can be converted by data movement using the COPY, or DUMP and RESTORE functions. It is possible and quite common to convert data in place, without the need for movement, by using the CONVERTV command.

See 5.4, “Converting from non-SMS to SMS” on page 298 for details.

The CONVERTV command allows the following tasks:

- ▶ The preparation of a volume for conversion by preventing new data set allocations
- ▶ Conversion simulation, to test for any data sets that do not match the expected criteria

- ▶ Actual conversion of a volume to SMS management
- ▶ Converting a volume from SMS management, for example, as part of a Disaster Recovery scenario

CONVERTV typically executes much faster than the COPY, or DUMP and RESTORE alternatives.

The advantage of using COPY, or DUMP and RESTORE are as follows:

- ▶ Non-SMS data sets can be migrated to SMS managed volumes with a new architectural design.
- ▶ SMS volume infrastructure requires VTOC, VTOC index, and VVDS data sets. These data sets might not be large enough on the original volumes, or might have errors that negate the conversion in place on the volume. The VVDS for example has additional information added on the original volume during the CONVERTV execution.

## DFSMSHsm

The hierarchical storage management (HSM)<sup>4</sup> component provides availability and space management functions. For more information, see the following resources:

- ▶ *DFSMSHsm Implementation and customizing Guide*, SC35-0418
- ▶ *DFSMSHsm Managing Your Own Data*, SC35-0420
- ▶ *DFSMSHsm Storage Administration*, SC35-0421

HSM improves productivity by making the most efficient use of disk storage, primarily by making better use of the primary volumes. It performs both availability management and space management automatically, periodically, and by issuing specific commands when manual operations are appropriate. Volumes can be:

- ▶ Managed by SMS. In this case the storage group definitions controls HSM-initiated automatic functions, depending upon the appropriate Management Class of the data set.
- ▶ Managed by HSM. These volumes are commonly known as primary or level 0. In this case each volume is defined to HSM individually by the ADDVOL parameter and governed accordingly.
- ▶ Owned by HSM. These incorporate migration level 1 (ML1), migration level 2 (ML2), backup, dump and aggregate volumes, and are a combination of disk and tape. Alternate tape copies can be made of ML2 and backup tapes for off-site storage.

Also, spill volumes can be generated from backups and ML2 volumes to improve tape usage efficiency.

---

<sup>4</sup> Other vendor products exist that have similar capabilities, and can be used in place of HSM, although they can restrict full exploitation of ISMF.

Figure 4-3 shows the relationship between the main components of the HSM environment.

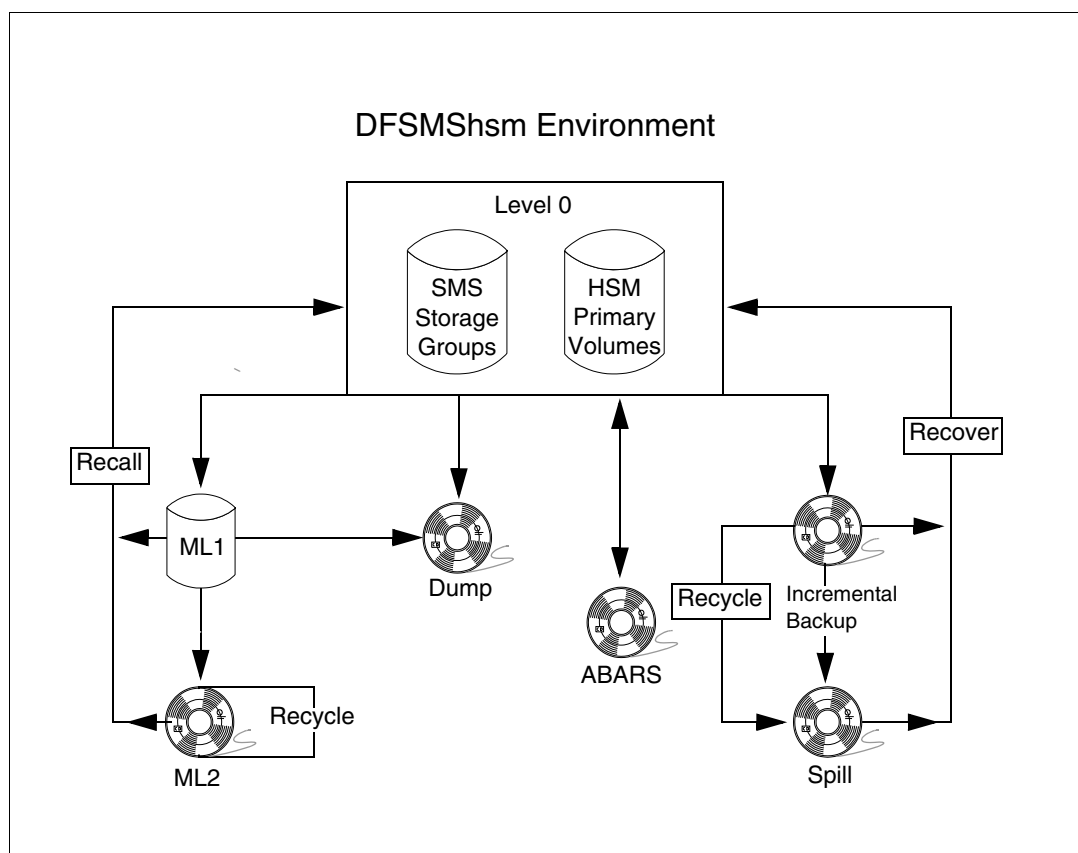


Figure 4-3 DFSMSHsm components

### Space management

On a daily basis, HSM performs automatic space management, depending upon whether the volume is part of an SMS storage group or managed individually by HSM. Space management consists of the following functions:

#### ► Migration

Migration is the process of moving eligible, inactive, cataloged data sets to either ML1 or ML2 volumes, from primary volumes (Level 0). This migration is determined by either the Management Class for SMS managed data sets, or set by the ADDVOL parameter for HSM managed. It can also be controlled in combination with volume thresholds set by the storage administrator. Data sets may be migrated to ML1 (normally disk)<sup>5</sup>, after a period of inactivity, and then onto ML2 (tape) following a further period of non-usage. It is feasible, and maybe more appropriate in certain cases, to migrate directly to ML2. Additionally, there is an *interval migration* process which can be used to free space on volumes during times of high activity.

#### ► Expiration processing

This function is based on the inactivity of data sets. For HSM-managed volumes, all data sets are treated in the same way on each volume. For SMS-managed volumes, expiration of a data set is determined by Management Class attributes for that data set.

<sup>5</sup> Very large data sets, in excess of 64,000 tracks, cannot be migrated to disk: they must be migrated to migration level 2 tape.

- Release of unused space

HSM can release over-allocated space of a data set for both SMS managed and non-SMS managed data sets. This includes LDSs that are used for DB2 data sets that are created with Extended Format (EF) and without the Guaranteed Space attribute enabled.

- Recall

Two ways of using recall are available:

- Automatically retrieving a data set when a user or task attempts to access it
- When the HRECALL command is issued

All recalls are filtered; if the data set is SMS-managed, then SMS controls the volume selection. If the data set is non-SMS, HSM directs the volume allocation. However, a possibility exists for the storage administrator to control a recall and place it on an appropriate volume or storage group if required. Recall of data sets that have the Guaranteed Space attribute enabled can recall the data set back to the volume it was migrated from.

If DB2 archive logs are migrated and you know that one is required, you can execute the HRECALL command to manually recall the data set to avoid DB2 waiting for the completion of the recall.

### ***Availability management***

The purpose of availability management is to provide backup copies of data sets for recovery scenarios. HSM can then restore the volumes and recover the data sets when they are needed. Note the following terminology:

- Incremental backups

This process takes a copy of a data set, depending on whether it has changed (open for output; browse is not sufficient), since its last backup. For SMS- managed volumes, HSM performs the backup according to the attributes of the Management Class of the individual data set. For non-SMS managed volumes, HSM performs the backup according to the ADDVOL definition.

- Full volume dumps

A full volume dump backs up all data sets on a given volume, by invoking DSS. HSM Dump Classes exist which describe how often the process is activated, for example, daily, weekly, monthly, quarterly or annually.

- Aggregate backup

Aggregate backup and recovery support (ABARS) is the process of backing up user defined groups of data sets that are business critical, to enable recovery should a scenario arise. Data sets copied can reside on and will be copied to disk, tape, or DFSMSHsm volume.

- Recovery

Recovery either can be at the individual data set level or can physically restore a full volume. Recovery is applicable for both SMS and non-SMS managed data sets.

Note that full exploitation of this component requires the use of the DSS and other components of DFSMS.

- FlashCopy

FlashCopy is executed by DFSMSHsm to provide fast backup of volumes or data sets. Backups are taken in two stages: logical and physical. From a DB2 perspective, logical copies are extremely fast, often taking less than a minute for hundreds, even thousands of volumes. For a large number of volumes, however, the physical copy can take hours to complete.

One key difference between the RVA and ESS/DS8000 environments is that the RVA used SnapShot technology that snapped pointers only and not real data, and FlashCopy copies real data. When using RVA technology, if a snapshot was taken and then application failure occurred shortly after, a snapshot back could immediately be executed. For the Enterprise Storage Server (ESS) and DS8000, a flash back requires that the physical copy be completed first, which could take a significant amount of time.

## DFSMSrmm

The Removable Media Manager (RMM)<sup>6</sup> component is the IBM tape management system for z/OS platforms. It provides a simple and flexible tape management environment, with support for all tape technologies, including IBM automated tape libraries, manual tapes, and other tape libraries, and volume and data-set-level management.

As part of DFSMS, RMM is completely integrated into the IBM storage management strategy. This integration allows for easier installation and maintenance, and standard interfaces with other systems components, such as DFSMSdfp and DFSMSHsm.

Tapes are an effective media for storing many types of data, especially backup and archive copies of disk data sets. However, tapes must be mounted to be used, and the capacity of tapes is often not fully used. DFSMS/MVS can be used to assist in more effective use of tape resources.

With SMS, a group of disks can be defined to act as a buffer for tape drives, and allow HSM to manage writing the tape volumes and data sets on those volumes. DFSMS/MVS permits the use of system managed tape volumes, and RMM can be used to manage the inventory of system managed and non system managed tapes.

For further information about this topic, see the following resources:

- ▶ *z/OS V1R11.0 DFSMSrmm Implementation and customizing Guide*, SC26-7405
- ▶ *z/OS V1R11.0 DFSMSrmm Guide and Reference*, SC26-7404

## Tivoli OMEGAMON XE for Storage on z/OS

Tivoli OMEGAMON XE for Storage on z/OS is one of the supporting products of DFSMS/MVS and is a separately orderable feature. It provides data analysis that assists in improving storage usage and reducing storage costs. It offers the following benefits:

- ▶ Provides detailed performance and availability information (data set, volume, control unit, channels).
- ▶ Provides DFHSM information and administration functions such as online HSM / DFDSS toolkit.
- ▶ Provides offline storage information (tape).
- ▶ Provides an application view (view all data sets an address space is touching).
- ▶ Provides logical volume support of all mainframe storage devices (specific physical device support for only IBM and specific vendors).
- ▶ Generates JCL for storage management tasks.

Details are available at the following web page:

<http://www.ibm.com/software/tivoli/products/omegamon-xe-storage/>

---

<sup>6</sup> Other Vendor products exist which have similar capabilities, and can be used in place of RMM, although this may restrict full exploitation of ISMF.

## SMF records type 42 subtype 6

DFSMS statistics and configuration records are recorded in SMF record type 42.

The type 42, subtype 6, SMF record provides information about data set level performance. DFSMS must be active, but the data set does not need to be SMS managed. Two events cause this record to be generated: data-set close-time and each type 30 interval record being written.

You can use OMEGAMON XE for Storage, or any SMF specialized package to format and display this useful record<sup>7</sup>. For instance, you can start by looking at the list of the first 20 data sets in terms of activity rate at the specified interval and verify that the most accessed DB2 data sets are performing as expected both in terms of I/O and usage of DB2 buffer pools.

Also, access to critical data sets can be tracked periodically by data set name. Their performance can be mapped against the IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS (OMEGAMON PE) accounting to determine detailed characteristics of the I/O executed, and to verify cache utilization.

### 4.1.3 Benefits

The following list summarizes the benefits of SMS:

- Simplified data allocation

SMS enables users to simplify their data allocations. Without using SMS, a user would have to specify the unit and volume on which the system should allocate the data set. In addition, space requirements would need to be calculated and coded for the data set. With SMS, users can let the system select the unit, volume and space allocation. The user therefore, does not need to know anything about the physical characteristics of the devices in the installation. This process was simplified with the use of PAVs, and when necessary the SMS Separation Profile.

- Improved allocation control

Free space requirements can be set using SMS across a set of disk volumes. The system automatically places data on a volume containing adequate free space.

- Improved performance

SMS can assist in improving disk I/O performance, and at the same time reduce the need for manual tuning by defining performance goals for each class of data. Cache statistics, recorded in System Management Facilities (SMF) in conjunction with OMEGAMON XE for Storage, can be used to assist in evaluating performance. Sequential data set performance can be improved by using extended sequential data sets. The DFSMS environment makes the most effective use of the caching abilities of disk technology.

- Automated disk space management

SMS has the facility to automatically reclaim space which is allocated to old and unused data sets. Policies can be defined that determine how long an unused data set are allowed to reside on level 0 volumes (active data). Redundant or expired data can be removed by the process of migration to other volumes (disk or tape), or the data can be deleted. Allocated but unused space can be automatically released, which is then available for new allocations and active data sets.

---

<sup>7</sup> A sample parsing routine is available at:  
[http://www.ibm.com/developerworks/exchange/dw\\_entryView.jspa?externalID=531&categoryID=33](http://www.ibm.com/developerworks/exchange/dw_entryView.jspa?externalID=531&categoryID=33)



- ▶ Improved data availability management

With SMS, separate backup requirements can be provided for data residing on the same primary volume. Therefore, all data on a single volume can be treated independently. The HSM component can be used to automatically back up data. The ABARS facility can be used to group data sets into logical components, so that the group is backed up at the same time, allowing for recovery of an application.

- ▶ Simplified data movement

SMS permits the movement of data to new volumes without the necessity for users to amend their JCL. Because users in a DFSMS environment do not have to specify the unit and volume that contains their data, whether their data resides on a specific volume or device does not matter. Simplified data movement allows the replacement of old devices with minimum intervention from the user.

System-determined block sizes can be utilized to automatically reblock sequential and partitioned data sets that can be reblocked.

## 4.2 Storage management with DFSMS

This section can help familiarize the DB2 administrator with the functionality of SMS and contains the following topics:

- ▶ Introduction
- ▶ Automatic class selection (ACS) routines
- ▶ SMS classes
- ▶ Naming standards

For further information, see the following publications:

- ▶ *z/OS V1R10.0-V1R11.0 DFSMS Introduction*, SC26-7397
- ▶ *z/OS V1R11.0 DFSMS Implementing System-Managed Storage*, SC26-7407
- ▶ *z/OS V1R11.0 DFSMS Managing Catalogs*, SC26-7409
- ▶ *z/OS V1R11.0 DFSMS Using the Interactive Storage Management Facility*, SC26-7411
- ▶ *z/OS V1R11.0 DFSMS Using the New Functions*, SC26-7473
- ▶ *z/OS V1R11.0 DFSMS Using Data Sets*, SC26-7410

### 4.2.1 Introduction

SMS manages an installation's data and storage requirements according to the storage management policy in use. The full SMS configuration requires a combination of ISMF panels and Automated Class Selection (ACS) routines. Using ISMF, the storage administrator defines the following items as an installation's storage management policy in an SMS configuration:

- ▶ Base configuration
- ▶ Class and storage group definitions

## **Base configuration**

The base configuration contains defaults and identifies the systems that are SMS-managed. The information is stored in SMS control data sets, which are VSAM linear data sets:

- ▶ Source control data set (SCDS)
- ▶ Active control data set (ACDS)
- ▶ Communications data set (COMMDS)

### ***Source control data set (SCDS)***

The SCDS contains the information that defines a single storage management policy, called an SMS configuration. More than one SCDS can be defined, but only one can be used to activate a configuration at any given time.

### ***Active control data set (ACDS)***

The ACDS contains the SCDS that has been activated to control the storage management policy for the installation. When a configuration is activated, SMS copies the existing configuration from the specified SCDS into the ACDS. While SMS uses the ACDS, the storage administrator can continue to create copies of the ACDS, modify, and define a new SCDS if desired.

### ***Communications data set (COMMDS)***

The COMMDS holds the name of the ACDS and provides communication between SMS systems in a multisystem environment. The COMMDS also contains statistics on the SMS, and MVS status for each SMS volume, including space.

## **Separation profile**

Storage administrators can supply the name of a data set that houses the separation profile, which allows the following tasks:

- ▶ Separation of data sets behind different disk controllers. For example, you separate the software dual copied active log and boot strap data sets behind different disk controllers for greater availability.
- ▶ Separation of data sets on volumes (as opposed to controllers) causing hot spots. For example, if you know that partition one and two must never reside on the same volume, you can add an entry in the separation profile to place them on different volumes. Volume and extent pool separation requires z/OS 1.11. Consider that in today's world, our volumes are logical, and several logical volumes reside on one physical disk. Disk controllers now house a huge amount of space. If only one disk controller exists, the data set on the same volume technique is also eligible, however use special care for data sets such as the active and boot strap data sets. If you cannot separate the active log and boot strap data sets onto different disk controllers, make sure they are allocated to different logical volumes, and physical volumes to. Depending on the outage, losing volumes even with RAID 5, 6, or 10 devices is possible. DB2 systems programmers should work with storage administrators so that important logical volumes do not reside on the same physical volume.

### ***Defaults set in the base configuration***

The defaults are as follows:

- ▶ Default Management Class
- ▶ Default Unit
- ▶ Default Device Geometry
  - Bytes/track
  - Tracks/cylinder

## Class and storage group definitions

The storage management policies are defined to the system by use of classes. Data sets have classes assigned to them. These are *Data Class*, *Storage Class*, and *Management Class*, and they determine the volume allocation which forms the *storage group*. The classes manage the data sets, and the storage groups manage the volumes on which the data sets reside. SMS allocations go through the same order, Data Class, followed by Storage Class, followed by Management Class, and finally storage group.

Figure 4-4 shows the steps taken by an installation to implement an active SMS configuration.

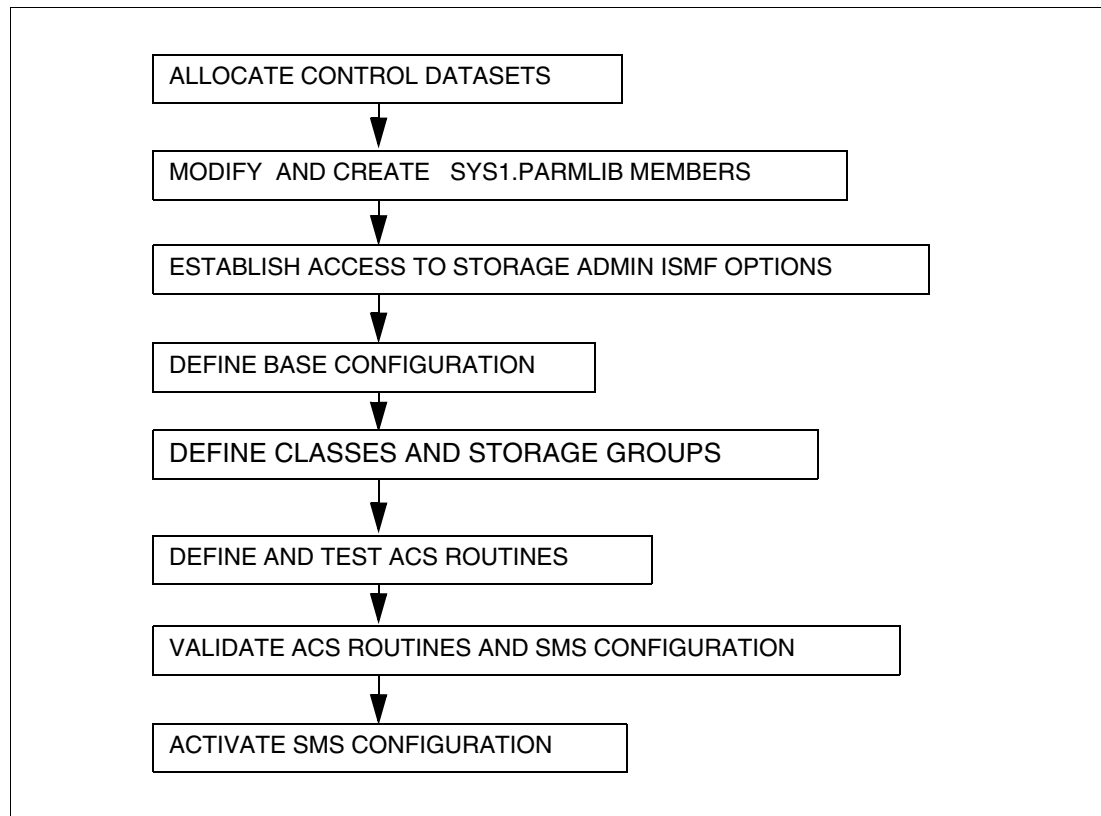


Figure 4-4 Implementing an SMS configuration

### 4.2.2 Automatic class selection (ACS) routines

ACS routines are the mechanism for assigning SMS classes and storage groups. They determine the placement of all new data set allocations, plus allocations involving the copying, moving, recalling, recovering, and converting of data sets. ACS routines are written in a high level REXX style programming language. If SMS is activated, all new data set allocations are subject to automatic class selection.

Four ACS routines are available. Aggregate groups also exist, but for the purposes of this book, are mentioned only where relevant. ACS routines are executed in the following order:

1. Data Class: data definition parameters
2. Storage Class: use of Guaranteed Space, use of multitiered storage groups, and number of stripes for data sets
3. Management Class: migration, backup, and retention attributes
4. Storage Group: candidate allocation volumes

Because data set allocations, whether dynamic or through JCL, are processed through ACS routines, installation standards can be enforced on those allocations on both SMS and non-SMS managed volumes. Non-SMS managed data sets can use some, but not all Data Class specifications. For example, non-SMS managed data sets can take advantage of the Data Class DCB related attributes as defaults without coding additional DCB information. Other Data Class specifications, such as using Extended Addressability or exceeding the 255 extent limit require the data set be SMS managed. Non SMS managed data sets only execute the Data Class portion and will exit out of the Storage Class, not continue down the path to the Management Class or Storage Group. ACS routines permit user defined specifications for Data, Storage, and Management Classes, and requests for specific volumes to be overridden, thereby offering more control of where data sets are positioned within the system. Specific volumes can only be requested with the use of Guaranteed Space in the Storage Class. Some storage administrators do not allow users to override the classes or request a specific volume.

For a data set to qualify for SMS management, it must satisfy the Storage Class component. If a *valid* Storage class is assigned, then the data set is managed by SMS, and proceeds down to the Management Class and Storage Group routines before being allocated on a SMS managed volume. If a *null* Storage class is assigned, the data set exits from the process, is not managed by SMS, and is allocated on a non-SMS volume.

Figure 4-5 shows the execution process for defining data sets in a z/OS system (with SMS active).

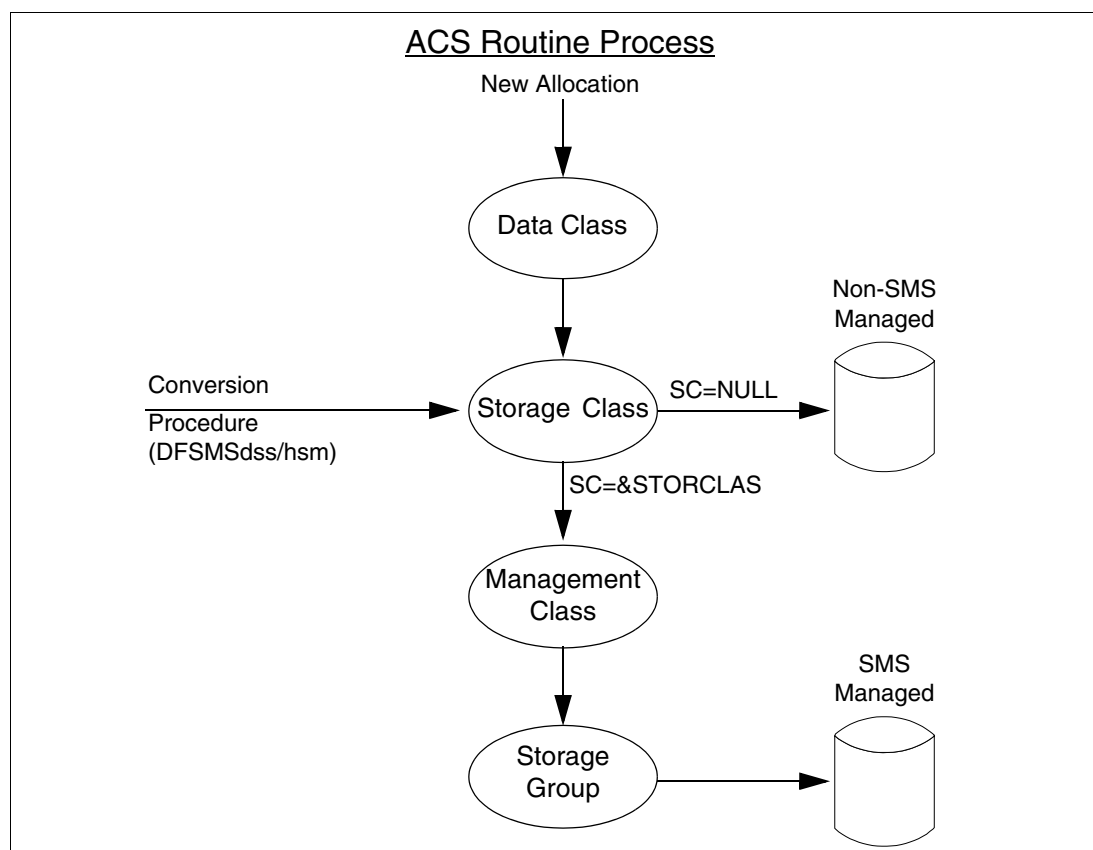


Figure 4-5 ACS routine execution process

### 4.2.3 SMS classes

The storage administrator uses ISMF to specify an ACS routine for each of the three types of classes and one to assign the storage groups. These routines, used together with the Data Class, Storage Class, Management Class, Storage Group definitions, and the base configuration, define an installation's SMS configuration.

Figure 4-6 shows the relationship of each of the four constructs that make up the SMS ACS routine environment:

- ▶ Data class
- ▶ Storage class
- ▶ Management class
- ▶ Storage group

This section describes each of the constructs in more detail.

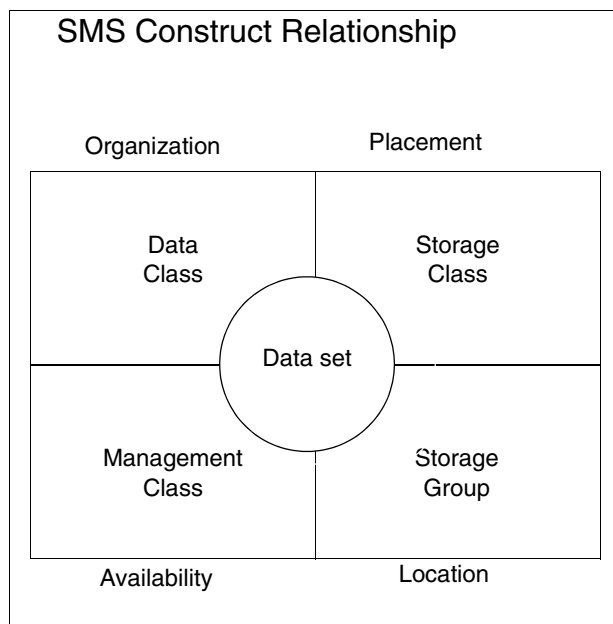


Figure 4-6 SMS Construct Relationship

#### Data class

Formerly, when allocating new data sets, users were required to specify a full set of attributes. Even for multiple allocations, repetitive coding was required. With the introduction of SMS, this process is now simplified, and also helps to enforce standards by use of the Data Class.

A Data Class is a named list of data set allocation and space attributes that SMS assigns to a data set when it is created. It contains associated default values set by the storage administrator. Data Classes can be assigned implicitly through the ACS routines or explicitly by using the following items:

- ▶ DB2 STOGROUP statement with keyword DATACLAS
- ▶ JCL statement

The user only has to specify the relevant Data Class keyword such as DATACLAS=DCLIB

- ▶ TSO/E ALLOCATE command DATACLAS(DCLIB)

- ▶ IDCAMS ALLOCATE and DEFINE commands
- ▶ ISPF/PDF Data set allocation panel

When specified, the Data Class allocates a data set in a single operation. Any disk data set can be allocated with a Data Class whether managed by SMS or not. Some storage administrators do not allow users to override any of the classes.

User defined allocations take precedence over default Data Classes. For example, if a Data Class specifies an LRECL of 80 bytes, and the JCL allocation specifies an LRECL of 100 bytes, and then 100 bytes are allocated. If the Data Class is altered by the storage administrator, attributes previously allocated by the class remains unchanged. Alterations are only be honored for *new* allocations.

### ***Planning for implementation***

To identify and reference a particular Data Class, a unique 1 - 8 character name is used, for example, DCDBKSDS.

For each group of data sets that have similar attributes, a Data Class can exist, but is not mandatory. An example of where it can be used is with DB2 user-managed table spaces, because they have identical allocation characteristics.

Prior to the definition of Data Classes, perform an analysis of common data types. For the analysis, decide whether to use ACS routines only for their allocation, or allow users (in this case, the DBA) to assign them also. There may be a requirement to standardize naming conventions, and agree upon default space allocations.

Attributes include many of the data set characteristics specified on JCL statements, and IDCAMS DEFINE statements. Only those applicable to a particular data set type should be coded, all others should be left blank.

Storage administrators have the ability to override user requested space attributes and replace them with those specified in ISMF for the Data Class.

The Data Class provides many options for data sets:

- ▶ DCB-related information
- ▶ Space-related information
- ▶ VSAM-related attributes, such as percentage of free space for the CI, CA, share options, and others, and use of buffering
- ▶ Retention period or expiration date
- ▶ Creating the data set volume (single or multi-volume), a follows:
  - DB2-managed data sets should not request more than one volume for Volume Count or Dynamic Volume Count. At end-of-volume, DB2 automatically adds a volume, if available, up to the DFSMSdfp limit of 59 volumes. The exception to this is when using DB2 storage groups without the VOLUMES parameter. In this case, SMS is in control and does not allow DB2 to use its normal end-of-volume (EOV) mechanism to add additional volumes.
  - User-managed data sets such as the DB2 catalog and directory can benefit by having values greater than 1 for Volume Count or Dynamic Volume Count. Although Volume Count places an asterisk (\*) for candidate volumes, Dynamic Volume Count does not. For example, you might want to have the storage administrator allocate the DB2 catalog and directory with a special Data Class assigned with a Dynamic Volume Count of 2, which would allow these data sets to extend to an additional volume if the first volume is full.

Volume Count and Dynamic Volume Count can be assigned together. For example, Volume Count has a value of 3, and Dynamic Volume Count has a value of 5. When the data set is first allocated on one volume, LISTCAT will show the VOLSER for the first volume and two “\*” for VOLSER, one for each candidate volume. When three volumes have been used and the fourth is allocated, LISTCAT shows the VOLSER for all four volumes and no entries for candidate volumes. When all five volumes have been exhausted, the data set can no longer be extended to a new volume.

- DB2 10 for z/OS (now announced) makes the DB2 catalog and directory a mandatory requirement to be SMS and DB2 managed.
- Image copy data sets that are required to be multi-volume can be set to a specific number of volumes.
- ▶ Whether extended format VSAM data sets, without the Storage Class attribute of Guaranteed Space, when allocating to subsequent volumes can take a primary or secondary allocation.
- ▶ Allow data sets to be Extended Format, Extended Addressable, or both. From a DB2 perspective, storage administrators can assign EF, EA, or both:
  - EF and EA are required for objects with a DSSIZE greater than 4 GB. These object types are partitioned, LOB, and XML. Simple and segmented table spaces cannot exceed 2 GB for each data set.
  - EF and the Storage Class use of SDR are required for VSAM striped data sets.
  - EF managed data sets contain a 32- byte suffix for each physical block. The 32 bytes are not visible and do not show up in a LISTCAT output or when examining the VSAM buffers. If you create a data set with EF enabled and one without, then load the same amount of data, you will almost never see a difference in the data set size.
  - Other file types besides enabling EF that can be specified are HFS, PDS, PDSE, and large format sequential data sets.
- ▶ Space Constraint Relief (SCR):
  - Bypasses the 5-extent rule where an allocation must complete within five extents. Although by default, the value is blank, adding a value, even 0, can bypass the DFSMSdfp 5-extent rule.
  - Provides the ability to reduce an allocation by a specified percent and redrive the allocation to determine if the lower value can be accommodated. For example, the storage administrator enters a value of 10. You have requested an allocation for 1,000 cylinders, however the space is not available on any eligible volume. SMS redrives the request, reducing the allocation request by 10%. In this case, the new request is for 900 cylinders (1,000 cylinders is 10%). The allocation is successful if any eligible volume has at least 900 cylinders available.
- ▶ Specific types of data sets are compressed or compacted on disk or tape. Data can use disk compression when allocated to the RVA, however disk compression is not available on the ESS or DS8000 devices.
- ▶ Tape-related specifications
- ▶ Encryption management
- ▶ Support of extended attributes DSCBs, primarily for EAV devices
- ▶ The ability to bypass the DFSMSdfp 255 extent rule. The rule in place states that a data set can have 123 extents per volume, and span up to 59 volumes with the maximum of 255 extents. The 255 extents part of the rule was eliminated; the new formula is 123 extents per volume multiplied by 59 volumes equals 7,257 extents. This is an architectural maximum. For example if your storage group only has 10 volumes, the limitation drops down to 123 extents per volume multiplied by 10 volumes equals 1,230 extents. There is a

good chance that not all data sets will reach 123 extents per volume, so the amount of allowable extents would be lower. By default, the 255 extent rule is still in place and if this option is desired, it must be overridden in the ISMF Data Class panel. Although this change can allow for greater availability by not limiting the number of extents to 255, much larger number of extents may cause performance related problems. Extending the 255 extent limit is an availability enhancement, not a performance one. Data sets must be SMS managed to bypass the 255 extent rule.

## Storage class

Prior to SMS, critical and important data sets that required improved performance or availability were allocated to specific volumes manually. Data sets that required low response times were placed on low activity volumes, where caching was available. SMS allows the separation of performance and service level of data sets by use of the Storage Class.

A Storage Class construct details the intended performance characteristics required for a data set assigned to a given class. The response times set for each Storage Class are target response times for the disk controller to achieve when processing an I/O request. It decides whether the volume should be chosen by the user or by SMS. The assignment of a Storage Class does not guarantee its performance objective, but SMS selects a volume that offers performance as close as possible. Only SMS-managed data sets use Storage Classes. Changes in a Storage Class applies to the data sets that are already using that class.

Storage Class performance characteristics used to be an important part of SMS. When the SMS was first introduced, disk technology was different than it is today. Cache was optional and speed of various disks varied greatly. The Storage Class addressed these issues based on cache and disk performance. Today, cache is always used and performance varies less. Generally, values for the performance portion are no longer entered, except for such requirements as SDR for striping.

Objects that are not managed by SMS have the ACS routine for the Storage Class set to null. When set to null, SMS does not follow the path to the Management Class or Storage Class, rather the data set will be non-SMS-managed.

Storage Classes can be assigned when authorized by the storage administrators, implicitly through the ACS routines, or explicitly by using the following items:

- ▶ DB2 STOGROUP statement with keyword STORCLAS
- ▶ JCL statement
- The user only needs to specify the relevant Data Class keyword, such as STORCLAS=SCDBGS.
- ▶ DSS COPY and RESTORE commands
- ▶ TSO/E ALLOCATE command such as STORCLAS(SCDBGS)
- ▶ IDCAMS ALLOCATE and DEFINE commands
- ▶ ISPF/PDF data set allocation panel

Unlike the Data Class, users *cannot* override individual attribute values when allocating data sets.

## Planning for implementation

For each group of data sets that have similar performance objectives, a Storage Class can exist. To identify and reference a particular Storage Class, use a unique name (using 1 - 8 characters), for example, SCDBFAST. Again, with the advent of newer devices, the performance objectives are generally no longer used.



Consideration whether a user is authorized to select a specific volume within a storage group, which is governed by the *Guaranteed Space* parameter through an arrangement between the storage administrator and the DBA. Guaranteed Space is as it sounds, it guarantees the value of the space requested. For this reason, Guaranteed Space cannot be use with the Data Class Space Constraint Relief (SCR). If you ask for 1,000 cylinders with Guaranteed Space enabled, 1,000 cylinders are required. Another name for Guaranteed Space might be guaranteed VOLSER.

- ▶ If you have a storage group with 100 volumes, you have Guaranteed Space enabled, you point to a specific VOLSER, only that VOLSER is eligible for allocation.
- ▶ When Guaranteed Space is not enabled and a you ask for a specific VOLSER, all 100 volumes are eligible and the allocation may or may not reside on the volume requested.
- ▶ When Guaranteed Space is not enabled and a you do not request a specific VOLSER, all 100 volumes are eligible for allocation.
- ▶ When a DB2 STOGROUP has a non-SMS-managed VOLSER, but the ACS routines allow for the data set to be SMS-managed, the data set will be SMS-managed despite the status of the volume in the DB2 STOGROUP. The allocation is on one of the 100 volumes when Guaranteed Space is not enabled.

Guaranteed Space does have other drawbacks too. When allocating onto multiple volumes, the primary is propagated to the other volumes and not the secondary. If you have a DB2-managed data set with 1,000 primary cylinders and 100 secondary cylinders, when the data set extends to the second volume the allocation will be 1,000 cylinders on the second volume, not 100 as with non-Guaranteed Space data sets. This way can be a bigger problem when allocating a multi-volume data set. Note the following information:

- ▶ APAR PK83108 resolves some of the problem when the object has Guaranteed Space enabled with specific entries in the DB2 STOGROUP. In this case, if a volume in the DB2 STOGROUP does not have enough space, it will redrive the request onto the next volume.
- ▶ Guaranteed Space can also be a problem with archive log data sets, therefore DSNZPARM SVOLARC was supplied. If set to YES, it tells DB2 to put a unit count of one in the new data set allocation request. Without this parameter DB2 could request a very large primary allocation on up to 15 volumes at the same time. This approach might cause severe space related problems for the archive log data sets when not enough space can be found.

An example of the use of this parameter is in the allocation of the *Active logs and BSDS*, where these data sets have critical availability requirements. The DBA should be allowed to allocate them on specific volumes, which is especially important for software dual logging capability, to ensure that the logs are allocated on separate volumes. After being defined, these data sets are rarely redefined. Dual software-copied data sets must be allocated on separate disk controllers when possible. When only one disk controller can be used, special attention needs to be made to allocate the data sets not only on separate logical volumes, but on separate physical volumes, and behind separate Logical Sub System/Logical Control Unit (LSS/LCU) when possible.

An alternative to using Guaranteed Space is to use the Separation Profile, which is part of the base configuration. The Separation Profile can be used to perform the following tasks:

- ▶ Separate the specific data sets onto different disk controllers during allocation. For example, LOGCOPY1 and LOGCOPY2 data sets must be allocated on different disk controllers.
- ▶ Separate the specific data sets on different volumes to avoid hot spots. The volume separation, as opposed to the controller separation, requires z/OS 1.11.

In the vast majority of cases, do not use Guaranteed Space for such things as the DB2 catalog and directory, work file, or user data sets. The software duplexed versions of the active log and boot strap data sets must be directly allocated either by using Guaranteed Space or by using the Separation Profile.

The Storage Class provides several options for the data set:

- Performance objects for devices

This option is no longer in general use.

- Striping

Striping is a combination of the Data Class attribute of Extended Format (EF), enabled in the ISMF Data Class panel, and a value for the Sustained Data Rate (SDR) in the Storage Class ISMF panel. The SDR rate is based on a calculation relating to the number of stripes required. SDR rate values for striping are different for data sets that use Guaranteed Space and ones that do not.

To stripe DB2 LDSs, you must set ZPARM DSVCI=YES. You cannot stripe data sets with DSVCI=NO when the page size is greater than 4 KB. This is because the page size and CI size must be the same to avoid partial writes. Striping data sets with DSVCI=NO with pages greater than 4 KB will eventually result in 00C20111 error.

- Use of Guaranteed Space
- If a multitiered storage group is allowed to be used
- If a volume with PAV is required
- CF and lock structure names and weights

## Management class

Prior to SMS, DFHSM managed the data sets at volume level, applying a standard management criteria for all data sets on a given volume. Although this is still applicable for non-SMS managed data sets, with the introduction of SMS, the control is carried out at *data set level* by use of the Management Class.

The Management Class is *only* assigned if the Storage Class construct selects a valid Storage Class, as can be seen in Figure 4-5 on page 206. For each data set, it consists of attributes that determine the necessary control of the following items:

- Retention
- Backup
- Migration
- Expiration
- Management of generation data set groups (GDGs) and their data sets (GDSs)
- Space release
- ABARS management

When assigned to a data set, the Management Class expands on attributes previously specified by JCL, IDCAMS DEFINE and DFHSM commands.

If an attribute is altered on a particular Management Class, the change is applied to previously created data sets which have the same attribute, when the next management cycle commences. A default Management Class can be specified to cater to those groups of data sets that do not belong to a Management Class, thereby ensuring that all SMS managed data sets have a set level of availability.

## ***Planning for implementation***

A Management Class must be generated for each group of data sets that have similar availability requirements. To identify and reference a particular Management Class, use a unique name (of 1 - 8 characters), for example, MCDB21.

Before defining Management Classes, a number of criteria should be established by the storage administrator. Based upon this information, the storage administrator defines Management Classes that provide a centralized storage environment. This includes the decision on whether to allow users the ability to assign Management Classes explicitly by using JCL, and implicitly by the ACS routines.

Because most production database data has specialized backup and recovery requirements, be sure that standard DB2 utilities are used to perform backup and recovery. However, consider using DFSMSdss or DFSMSHsm's automatic backup services, supported by concurrent copy, to help with *point of consistency* backups.

Using HSM to manage most production databases is not advisable. Therefore, use a NOMIGRATE Management Class for this type of data. This prevents HSM space and availability management from operating. Specifically, AUTO BACKUP is set to NO so that HSM does not back up the data set, ADMIN OR USER COMMAND BACKUP is set to NONE to prevent manual backup, and expiration attributes are set to NOLIMIT to prevent data set deletion.

HSM can expire such objects as the archive log data sets and image copies, which is an alternative to using tape retention data sets.

Although production database data does receive automatic backup service, DFSMSdss can be set to run concurrent copy for production databases. ACCESSIBILITY may be set to CONTINUOUS for Storage Classes assigned to production databases to ensure that the data set is allocated behind a storage control with concurrent copy support. With newer devices, concurrent copy should be on for all IBM devices.

Testing or user databases that have less critical availability requirements can benefit from system management using HSM. Additionally, selected data types for production systems could also be effectively managed using HSM.

For DB2 environments, managing archive logs and image copies with HSM is possible. These data sets can be retained on primary devices for a short period of time, and then migrated directly to tape (ML2). Some customers allocate one archive log data set to a disk that DFSMSHsm sweeps every 24 hours and the other archive log data set swept hourly. Because data sets can be allocated onto empty tapes or added (with MOD status allocation) onto tapes that already have data, there is no guarantee the software duplexed pair will not eventually reside on the same tape. In this case, the tape snapping can cause a single point of failure.

To avoid a single point of failure, the object should be duplexed. Duplexing can be accomplished by having DFSMSHsm duplex tapes, mirroring the tape subsystem, and manually copying or transmitting data sets.

For non-production environments, various customers release unused space in the DB2 LDSs when allocated with EF and Guaranteed Space is not enabled.

The storage administrator and the DBA must agree on the requirements.

## Storage group

Prior to SMS, disk storage was maintained as individual volumes, requiring manual intervention to prevent volumes from filling up, and to prevent I/O bottlenecks. SMS significantly improves the management of disk by building on DFHSM capabilities to pool volumes together in storage groups. The benefit has been expanded by use of PAV and the SMS Separation Profile.

A *storage group* is a pool of disk volumes upon which SMS managed data sets are placed. These groups are normally transparent to users. A data set is placed on an appropriate volume within a storage group depending upon the Storage Class, volume status, storage group status, available free space, and channel utilization.

Allocations are done using a sophisticated algorithm. DB2 allows DB2 storage groups to have duplicate VOLSERS; the SMS storage group allows only a specific volume to belong to one storage group.

Each storage group has attributes that determine a range of characteristics for the volumes in that group. This includes backup, migration, and space thresholds.

A volume can belong to one of the following main storage group types:

- ▶ POOL
- ▶ COPY POOL
- ▶ BACKUP
- ▶ DUMMY
- ▶ VIO

Three other types also exist, although they are not as commonly used:

- ▶ OBJECT
- ▶ OBJECT BACKUP
- ▶ TAPE

Note the following information about storage groups:

- ▶ Cannot share a volume.
- ▶ Cannot share data sets unless extended storage groups are used.
- ▶ Cannot be different device types, but can be different emulated volume types. You cannot have a 3380 and a 3390 in the same storage group, but you can have a 3390 Model 3 and a Model 9 in the same one.
- ▶ Must contain whole volumes.
- ▶ Must contain volumes of the same device geometry.
- ▶ Can contain multi-volume data sets.
- ▶ Must contain a VTOC, VTOC index, and a VVDS.

SMS selects the volumes used for data set allocation by building a list of all volumes from the storage groups assigned by the ACS routine. Volumes are then either removed from further consideration or flagged as primary, secondary, or tertiary volumes. If no volumes meet the criteria, the request results in a rejected status.

Allocations take in consideration the channel utilization based on feedback from SRM (System Resources Manager). SRM does not keep long lasting historical data for trending, meaning that allocations are based on recent utilization numbers.

The volumes are assigned to separate groups depending on the following criteria:

- **Primary volumes**

Primary volumes must be online, below threshold, and meet all the specified criteria in the storage class. Both the volume status and storage group status are enabled. Volume selection starts from this list.

For EAS-eligible data sets on devices with cylinder-managed space, both the track-managed space and cylinder-managed space threshold, and the total volume space threshold are assessed to determine whether the volume gets placed on the primary volume list.

- **Secondary volumes**

Secondary volumes do not meet all the criteria for primary volumes. SMS selects from the secondary volumes if no primary volumes are available.

- **Tertiary volumes**

Volumes are classified as tertiary if the number of volumes in the storage group is less than the number of volumes that are requested. SMS selects from the tertiary volumes if no secondary volumes are available.

- **Rejected volumes**

Rejected volumes are those that do not meet the required specifications. They are not candidates for selection.

After the system selects the primary allocation volume, that volume's associated storage group is used to select any remaining volumes requested except when extended storage groups or overflow volumes are used.

### ***Planning for implementation***

Because of their critical status, unique storage groups must be assigned for DB2 production data sets. Appropriate groups should be defined by the storage administrator to prevent automatic migration (AUTO MIGRATE), which includes not releasing any unused space, and automatic backup (AUTO BACKUP). Customers must decide how their non-production databases will be used. Certain customers view their non-production environments as critical as their production environments; others classify the non-production environments as not nearly as important. Some customers choose to merge all non-production data, understanding the negative ramifications of this approach, including such issues as performance, and volume based backup and recovery.

DB2 allows the DBA to define a collection of volumes that DB2 uses to find space for new data set allocation, known as storage groups. When converting DB2 databases to SMS, and DB2 storage groups are used to manage DB2 database data, one way is to design the SMS storage groups so they are compatible with existing STOGROUP definitions.

When the conversion is complete, be sure SMS rather than DB2 is used to allocate databases. To allow SMS control over volume selection, define DB2 storage groups with VOLUMES(\*) value (specify only one asterisk).

Electing DB2 to select the volume requires assigning a Storage Class with Guaranteed Space. However, Guaranteed Space reduces the benefits of SMS allocation, so this approach is not recommended.

If you do choose to use specific volume assignments, additional manual space management must be performed. Free space must be managed for each individual volume to prevent failures during the initial allocation and extension. This approach generally requires more time for space management, and results in more space shortages. Generally, use Guaranteed

Space only for such objects as the active log and boot strap data sets. An alternative is to use the Separation Profile in the SMS base configuration.

To identify and reference a particular storage group, use a unique name (1 - 8 characters), for example, SGDBFAST.

The storage group can be used for various purposes. One purpose is the COPY POOL BACKUP for example, which allows for using FlashCopy.

Another common purpose is using Pool for allocating a pool of volumes and can be used for the following tasks:

- ▶ DFSMSHsm migration
- ▶ DFSMSHsm full volume dump
- ▶ DFSMSHsm incremental backup

Pool can also be used as follows:

- ▶ To specify whether an overflow volume is used
- ▶ To specify an Extend Storage Group if requested

Extended storage groups allow data sets that must be extended to do so to a separate storage group. For example, some customers have a table space pool and a separate index pool. This separation is not required when enough PAVs exist, such as with dynamic and Hyper PAV. In this case, if a table space is allocated to the table space pool and needs to extend and no volume meets the criteria and is therefore in a rejected status, the index pool can instead be used for the extend to avoid a failure. Overflow volumes can be used as an Extend Storage Group.

- ▶ To specify an associated Copy Pool Backup name
- ▶ To specify the HIGH value

This value is used as part of the allocation algorithm. For primary selection, a volume cannot be selected if it exceeds the HIGH threshold. This value generally increases as the model of disk emulated increases. 85% of a mod 3 device has much less space than 85% of a mod 9 device. HIGH is used to allow for primary allocations to grow and allow for secondary allocations, avoiding multi volume data sets when possible. A value of 99 allows for more multi-volume data sets; a lower value allows for less. Again, this is tied to the size of the emulated disk. Note the following information:

- The HIGH value is associated with the LOW value when using DFSMSHsm migration. During the migration process, after the HIGH value is exceeded DFSMSHsm will migrate data sets down to the LOW value. Not all HIGH values across all pools are the same even when all volumes have the same emulated disk types. For example, for a storage group using mod 3 device for DB2 user table spaces and indexes, you might want to have a HIGH value of 85%. For a storage group with archive logs that are swept hourly and not manually migrated through Automated Operations (AO), the HIGH value is generally low to trigger the migration even when a small amount of data exists.
- When the HIGH value has been exceeded for the storage group, the following SMS message is issued:

```
IGD17380I STORAGE GROUP (xxxxxxx) IS ESTIMATED AT nn% OF CAPACITY, WHICH  
EXCEEDS ITS HIGH ALLOCATION THRESHOLD OF nn%.
```

For DB2-related storage groups, IGD17380I is generally displayed in the DBM1 STC output and on the MVS console. AO must be in place to report on these messages for

the storage group housing the DB2 catalog and directory and the user table spaces and indexes. Prompt measures must be taken to avoid out of space failures.

- The LOW value tells DFSMSHsm migration how low of a percent the volume should remain. A higher value means that DFSMSHsm will require less work to migrate the eligible data sets, a lower number means more work and time for DFSMSHsm.
- ▶ For production environments, because DB2 table spaces and indexes are generally not migrated, the LOW value will have no effect for the DB2 user pool, however the HIGH value is still used. For production environments, LOW is still useful for such pools as when migrating data sets from a pool dedicated to archive logs and image copy data sets.
- ▶ For the newer Extended Addressable Volumes (EAV), the storage group can specify the Break Point Value (BPV) to determine when allocations should be placed in the cylinder managed portion of the EAV.

For further information about all SMS Class attributes and definitions, see *DFSMS/MVS DFSMSdfp Storage Administration Reference*, SC26-7402.

### ***Mapping devices to storage groups for performance***

VOLSERs are logical and tied to a VTOC. The actual devices are mapped to different physical locations on the disk box. The storage administrator needs to architect the disk box to avoid performance and outage related problems. Two logical volumes can reside on the same physical drive, losing more than one physical drive may cause a disk outage depending on the type of RAID utilized. Disk mapping must occur to prevent over utilizing any specific LSS/LCU \*Logical Sub System/Logical Control Unit) and prevent outages.

## **4.2.4 Naming standards**

To assist in the successful implementation of SMS, a requirement is that of generating and adhering to a constructive and meaningful naming standard policy. The more formal the policy, the easier it is to maintain the ACS routines.

These policies offer the following features:

- ▶ Simplify service-level assignments to data.
- ▶ Facilitate writing and maintaining ACS routines
- ▶ Allow data to be mixed in a system-managed environment while retaining separate management criteria
- ▶ Provide a filtering technique useful with many storage management products
- ▶ Simplify the data definition step of aggregate backup and recovery support

Most naming conventions are based on the high-level qualifier (HLQ) and low level qualifier (LLQ) of the data set name. Additional levels of qualifiers can be used to identify generation data sets and databases. They can also help users to identify their own data. An important point is that each installation has different naming conventions, and therefore requires careful planning.

DB2 systems generate their own data set names, so it is necessary to ensure that the storage administrator understands the implications, and is able to define a policy and build the ACS routines so they incorporate this feature. For example, if we want to separate the DB2 catalog and directory data sets, we know the outline of each qualifier and that the third qualifier will be DSNDB01 for the directory and DSNDB06 for the catalog. The ACS routine can be used with a combination of qualifiers and wild cards.

## 4.3 SMS examples for DB2 databases

The examples in this section show how SMS can be used to manage DB2 table spaces. These examples do not show all possibilities SMS offers to an installation. Each installation can review these examples and create those classes that best suit its environment. The examples shown here are extracted and adapted from *DFSMS/MVS Implementing System-Managed Storage*, SC26-7407.

### Naming convention

Storage administrators typically set up SMS classes and storage groups with a specific naming convention. The naming convention is for ease of management, rather than a restriction:

- ▶ Data Classes names start with DC.
- ▶ Storage Class names start with SC.
- ▶ Management Class names start with MC.
- ▶ Storage Group names start with SG.

For example, a Data Class can be defined and used as MYTEST and does not need to adhere to the DC names. An example of a Data Class name to enable EF and EA might be DCEFEA.

### 4.3.1 Using ISMF to display SMS constructs

A DB2 administrator can use ISMF to access and examine the various SMS constructs in the installation. A storage administrator uses ISMF to create and to manage the SMS constructs. Figure 4-8 on page 220 shows how to display the active Data Class DCDB2.

The options available on the DATA CLASS APPLICATION SELECTION panel depend on the authorization of the user. Only a user who is authorized to manage SMS constructs is allowed to define or alter them. Other users might only have options 1 (List) and 2 (Display) available. Various RACF profiles can tighten authorization for various panels.

Authorized users of ISMF have the option to change their ISMF profile. At ISMF startup, one panel is available for users and another panel with many more options for storage administrators.

The initial users panel, when you first use ISMF, is shown in Figure 4-1 on page 195. To change the view to storage administrator, select option **0** for ISMF Profile, and then **0 User Mode Selection**. The panel of Figure 4-2 on page 196 is displayed.

Enter **2** for storage administrator and press Enter. Press PF3 until you are out of ISMF. When you return ISMF, you see the expanded panel used by the storage administrator (Figure 4-7 on page 219).



```

ISMF PRIMARY OPTION MENU - z/OS DFSMS V1 R11
Selection or Command ==>

0 ISMF Profile           - Specify ISMF User Profile
1 Data Set               - Perform Functions Against Data Sets
2 Volume                 - Perform Functions Against Volumes
3 Management Class       - Specify Data Set Backup and Migration Criteria
4 Data Class              - Specify Data Set Allocation Parameters
5 Storage Class           - Specify Data Set Performance and Availability
6 Storage Group           - Specify Volume Names and Free Space Thresholds
7 Automatic Class Selection - Specify ACS Routines and Test Criteria
8 Control Data Set        - Specify System Names and Default Criteria
9 Aggregate Group         - Specify Data Set Recovery Parameters
10 Library Management     - Specify Library and Drive Configurations
11 Enhanced ACS Management - Perform Enhanced Test/Configuration Management
C Data Collection          - Process Data Collection Function
G Report Generation       - Create Storage Management Reports
L List                   - Perform Functions Against Saved ISMF Lists
P Copy Pool              - Specify Pool Storage Groups for Copies
R Removable Media Manager - Perform Functions Against Removable Media
X Exit                   - Terminate ISMF

```

Figure 4-7 Storage administrator panel

In user mode, notice that there is no option for the Storage Group. Other missing options are options 11 and C. These options allow you to easily collect data using DCOLLECT and run sample reports that convert the raw DCOLLECT output to useful readable reports. These reports are flexible and can be useful to help manage your DB2 volumes and data sets.

### 4.3.2 SMS data class

It is not unusual to find that a storage administrator has not assigned an SMS Data Class for any DB2 data sets. For a DB2 environment, the Data Class can be used in several ways:

- ▶ Instead of coding any common DCB information for like sequential files, a Data Class can be created that can also include space specifications.
- ▶ The DB2 catalog and directory are user-defined during the installation process using IDCAMS statements. Generally, when executing the DEFINE commands, only one volume is specified for a data set. Unlike DB2-managed data sets, when end-of-volume is reached and only one volume was specified, a DB2 error will result. Depending on which data set was unable to extend determines the extent of the problem. To avoid such a problem, specify VOLUME COUNT or DYNAMIC VOLUME COUNT.
- ▶ Extended Format (EF) data sets are used for the following reasons for DB2 data sets:
  - To assign data sets with a size greater than 4 GB. This assignment can be used for very large partitions, LOBS, and XML objects with a DSSIZE greater than 4 GB. EA, which is a subsection of EF, is required for this option. Both EF and EA must be enabled to create partitions, LOBS, and XML objects with a DSSIZE greater than 4 GB.
  - For VSAM striping of a data set. Heavily sequential processing can benefit from VSAM striping which also requires a value for the Sustained Data Rate (SDR) in the Storage Class.

- Bypass the DFP 5 extent rule for allocations.
- When space is not available on any volumes, redrive the space requests for the data set with a new request lowered by the specified percent. This option cannot be used with data sets that use the Storage Class attribute of Guaranteed Space. Guaranteed Space means just that, SMS honors the space requested, which must be the space allocated.
- Remove the 255 extent limit for the VSAM LDSs. By default, the 255 extent limit is in place, which can be a problem for availability, but can also be a problem when using Sliding Secondary with a DSSIZE of 16 or 64 GB. For 16 and 64 GB data sets, the algorithm assigns close to the full 255 extent limit to fill out the data set. Fragmentation can cause data sets to take additional extents which may cause a problem with Sliding Secondary. Data sets must be SMS-managed to bypass the 255 extent rule.

Users wanting to verify that a Data Class actually bypasses the 255 extents, can do so going to ISMF and displaying the Data Class. The ISMF Data Class has five panels. Assuming the information in the first four panels are correct, you view the fifth panel, which is shown in Figure 4-8.

```

DATA CLASS DISPLAY
Command ==>

CDS Name . . . : ACTIVE
Data Class Name : DB2EFEAX

Reuse . . . . . : NO
Initial Load . . . . . : RECOVERY
BWO . . . . . :
Log . . . . . :
Logstream Id . . . . . :
FRlog . . . . . :
RLS CF Cache Value . . . . . : ALL
RLS Above the 2-GB Bar . . . : NO
Extent Constraint Removal . : YES

```

Figure 4-8 Data Class Display, panel five of five

Verify that Extent Constraint Removal is set to YES. Then, execute LISTCAT or ISPF 3.4 and verify that the data sets you are interested in have this specific Data Class assigned. You must execute this against the cluster portion only, because the data portion does not display the Data Class. This means you must use DSNDDBC and not DSNDBD.

When reviewing LISTCAT and other display functions, such as ISPF 3.4 or ISMF, only the SMS classes are displayed, not the SMS storage group. The storage group for a specific data set is displayed only on specific conditions, such as for specific error conditions.

DB2 9 allows DB2 storage groups to be created or altered with any or all of the SMS classes. DB2 storage groups cannot specify SMS storage groups, only the classes. When an SMS class is used, the VOLUMES parameter is optional.

Note the following information:

- The rules of the CREATE STOGROUP in the *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854 indicate that SMS controls the volume selection if you omit the VOLUMES clause. This means that you can use CREATE STOGROUP and choose one or more SMS classes and omit the VOLUMES parameter. When this option is selected, DB2-managed

data sets do not extend to new volumes as when a VOLUMES clause entry is present. With VOLUMES (“\*”) clause, which allows SMS to control the volume allocation, when end of volume is reached, SMS chooses a new volume to extend to. When VOLUMES is omitted, the Data Class entry for Volume Count (VC) and Dynamic Volume Count (DVC) are in control. Typically VC is set to 1 (one) and DVC is null for DB2-managed data sets because of the way DB2 drives the extend operation for the next volume. In this case, when you omit VOLUMES and have the default VC=1 set and no DVC, extending to a new volume is not possible and fails the request. The VC=1 and no DVC settings tell SMS to allow only one volume, and extending to additional volumes is not possible. If customers prefer to omit VOLUMES then VC or DVC must be to the number of volumes desired. Consider that DFSMSdfp allows data sets to extend up to 59 volumes. If you choose a value lower than 59 for VC, DVC, or both, a data set can be extended only to that many volumes, lower than the DFSMSdfp default.

- ▶ When VOLUMES is specified, DB2-managed data sets do not use VC or DVC and they are ignored if specified in the Data Class. At EOV, DB2-managed data sets can then be allocated one volume at a time to the DFSMSdfp maximum of 59 volumes rather than the artificial limit restricted by VC, DVC, or both.

Assuming the storage administrator allows you to code an SMS class, this option provides the flexibility for allocations. Many storage administrators do not assign a Data Class for DB2 objects; others create all DB2 LDSs as EF/EA-enabled. Enabling EF/EA for all DB2 LDSs provides the flexibility to create data sets greater than 4 GB when required without intervention from the storage administrator. Various alternatives are as follows:

- ▶ DBA must notify the storage administrator of any new objects that are to be EF/EA-enabled, which will be added to the Data Class ACS routine. This option is the least flexible because it requires the storage administrator to update the ACS routines and SMS Control Data Set (CDS) every time a change is required. For many users, this requires a change control process that can take days or weeks to process.
- ▶ Creating or altering a DB2 STOGROUP with a Data Class specified is possible. For example, the storage administrator has created a Data Class called DB2EFEA that enables EF/EA, but does not assign a Data Class for DB2 LDSs. This Data Class is not assigned to any objects, it was created in the ISMF panels, but not referenced in the Data Class ACS routines. In this specific scenario, if you choose to create a new DB2 STOGROUP and not alter an existing one, the syntax would be similar to the syntax in Example 4-2.

*Example 4-2 Create a storage group syntax*

---

```
CREATE STOGROUP SUEFEA
  VCAT ARI DATACLAS DB2EFEA;

CREATE TABLESPACE RACHEL
  USING STOGROUP SUEFEA
  PRIQTY          48
  SECQTY          48
  LOCKSIZE        ANY
  CLOSE           NO
  BUFFERPOOL      BPO
  FREEPAGE        0
  PCTFREE         0;
```

---

After the table space is created, LISTCAT shows a Data Class for the cluster portion of the LDS as DB2EFEA.

Some storage administrators set up the ACS routines to work off a previous routine. For example, they might set up the Storage Class ACS routine to assign a specific Storage Class based on a specific Data Class name. Verify with your storage administrator before setting any SMS class in the DB2 STOGROUP, making sure the objects will still be properly assigned. As an example, we have created the previous data set, but we want to assign a special Storage Class for Guaranteed Space called DB2GS.

If the storage group ACS routine was written with a filter list based on data set name, your allocation should be successful:

```
FILTLIST DB2 INCLUDE(ARI.DSNDB%.**)
WHEN (&DSN = &DB2) SET &STORGRP = 'DB9A'
```

If the storage group ACS routine was written based on a Storage Class name passed, your allocation based on the storage group ACS routine would either, by luck be successful or allocate into the wrong storage group, or it might fail:

```
WHEN (&STORCLAS = 'DB9A')
SET &STORGRP = 'DB9A'
```

In the latter case, your DB2 STOGROUP is pointing to STORCLAS DB2GS and does not meet the criteria of the Storage Class name equal to DB9A.

### 4.3.3 SMS storage class

Storage administrators typically create SMS Storage Classes in a DB2 environment for the following reasons:

- ▶ To enable the Guaranteed Space attribute
- ▶ To allow for the use of VSAM striping whereby a data set is created with multiple stripes. This requires the SDR rate to be set and the Data Class EF enabled. This should only be used for heavily sequential objects.
- ▶ To allow multitiered storage groups to be enabled. Although this is actually a storage group function, it is enabled in the ISMF Storage Class panel.

To verify the Storage Class setting, select option **5** from the ISMF panel for Storage Class. List all Storage Classes or you can display a specific Storage Class.

Most of the entries in the ISMF panel for the Storage Class are related to older disks and are no longer used.

To illustrate how striping works, we take a table space that is heavily sequential:

- ▶ The table space has no Data Class assigned
- ▶ The table space has a standard Storage Class with no special features, including no use of Guaranteed Space or any value for SDR.

We want to convert it to a VSAM striped data set with three stripes, with no Guaranteed Space. The initial allocation was again, no Data Class assigned, and a Storage Class with no Guaranteed Space, and no SDR rate value. We can use the Data Class DB2EFEA in the previous example where EF and EA were both enabled. For striping, we require only EF, but the EA portion can be on also, however it is not required. We create a new Storage Class that has an SDR rate of 12 called DB9ASDR. Because the data set does not use Guaranteed Space, we divide the SDR rate by four with the result being the number of stripes, in this case  $12 \text{ SDR} / 4 = 3 \text{ stripes}$ . This means that the allocation is now be striped onto three volumes instead of one. See Example 4-3 on page 223.

*Example 4-3 Original allocation non-striped*

---

```

CREATE TABLESPACE JOHNITS3
    USING STOGROUP SYSDEFLT
    PRIQTY          7200
    SECQTY          1440
    LOCKSIZE        ANY
    CLOSE           NO
    BUFFERPOOL      BP0
    FREEPAGE        0
    PCTFREE         0;

LISTCAT after creating the table and inserting a row:
CLUSTER ----- DB9AU.DSNDBC.DSNDB04.JOHNITS3.I0001.A001
IN-CAT --- UCAT.DB9A
HISTORY
    DATASET-OWNER----PAOLOR9      CREATION-----2010.125
    RELEASE-----2              EXPIRATION-----0000.000
SMSDATA
    STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
    DATACLASS -----(NULL)      LBACKUP ---0000.000.0000

ATTRIBUTES
    KEYLEN-----0              AVGLRECL-----0
    BUFSPACE-----8192         CISIZE-----4096
    RKP-----0                MAXLRECL-----0
    EXCPEXIT----- (NULL)      CI/CA-----180
    SHROPTNS(3,3)    SPEED      UNIQUE          NOERASE      LINEAR
    NOWRITECHK      NOIMBED      NOREPLICAT
    UNORDERED      REUSE        NONSPANNED

EXTENTS-----1

ALLOCATION
    SPACE-TYPE-----CYLINDER      HI-A-RBA-----7372800
    SPACE-PRI-----10            HI-U-RBA-----7372800
    SPACE-SEC-----2

VOLUME
    VOLSER-----SBOX1Z        PHYREC-SIZE-----4096
    HI-A-RBA-----7372800     EXTENT-NUMBER-----1
    DEVTYPE-----X'3010200F'   PHYRECS/TRK-----12
    HI-U-RBA-----7372800     EXTENT-TYPE-----X'40'
    VOLFLAG-----PRIME        TRACKS/CA-----15
    EXTENTS:
    LOW-CCHH-----X'04A80000'   LOW-RBA-----0
    TRACKS-----150
    HIGH-CCHH-----X'04B1000E'  HIGH-RBA-----7372799

```

---

After the initial allocation, the table space is stopped. A new DB2 STOGROUP is created:

```

CREATE STOGROUP JOHNSTRI
    VCAT DB9AU DATACLAS DB2EFEA STORCLAS DB9ASDR;

```

The table space is altered to the new STOGROUP:

```
ALTER TABLESPACE JOHNNITS3 USING STOGROUP JOHNSTRI;
```

The table space is started, and then is reorganized (REORG).

Example 4-4 shows the output of LISCAT for the striped data set.

*Example 4-4 LISTCAT after the data set is striped*

---

```

CLUSTER ----- DB9AU.DSNDBC.DSNDB04.JOHNNITS3.I0001.A001
  IN-CAT --- UCAT.DB9A
  HISTORY
    DATASET-OWNER----PAOLOR9      CREATION-----2010.125
    RELEASE-----2              EXPIRATION-----0000.000
  SMSDATA
    STORAGECLASS ----DB9ASDR      MANAGEMENTCLASS---MCDB22
    DATACLASS  -----DB2EFEA    LBACKUP ---0000.000.0000

ATTRIBUTES
  KEYLEN-----0      AVGLRECL-----0
  BUFSPACE-----8192  CISIZE-----4096
  RKP-----0         MAXLRECL-----0
  EXCPEXIT----- (NULL)  CI/CA-----180
  STRIPE-COUNT-----3
  SHROPTNS (3,3)      SPEED      UNIQUE      NOERASE      LINEAR
  NOWRITECHK      NOIMBED      NOREPLICAT
  UNORDERED      REUSE      NONSPANNED      EXTENDED      EXT-ADDR

EXTENTS-----3

ALLOCATION
  SPACE-TYPE-----TRACK      HI-A-RBA-----7372800
  SPACE-PRI-----150      HI-U-RBA-----24576
  SPACE-SEC-----30

VOLUME
  VOLSER-----SBOX1Y      PHYREC-SIZE-----4096
  HI-A-RBA-----7372800      EXTENT-NUMBER-----1
  DEVTYPE-----X'3010200F'      PHYRECS/TRK-----12
  HI-U-RBA-----24576      EXTENT-TYPE-----X'40'
  VOLFLAG-----PRIME      TRACKS/CA-----5
  STRIPE-NUMBER-----1
  EXTENTS:
    LOW-CCHH-----X'045C0005'      LOW-RBA-----0
  TRACKS-----50
    HIGH-CCHH-----X'045F0009'      HIGH-RBA-----7372799
VOLUME
  VOLSER-----SBOX1X      PHYREC-SIZE-----4096
  HI-A-RBA-----7372800      EXTENT-NUMBER-----1
  DEVTYPE-----X'3010200F'      PHYRECS/TRK-----12
  HI-U-RBA-----0      EXTENT-TYPE-----X'00'
  VOLFLAG-----PRIME      TRACKS/CA-----5
  STRIPE-NUMBER-----2
  EXTENTS:
    LOW-CCHH-----X'046C0005'      LOW-RBA-----0
  TRACKS-----50

```

HIGH-CCHH-----X'046F0009'	HIGH-RBA-----7372799
VOLUME	
<b>VOLSER-----SBOX0G</b>	PHYREC-SIZE-----4096
HI-A-RBA-----7372800	EXTENT-NUMBER-----1
DEVTYPE-----X'3010200F'	PHYRECS/TRK-----12
HI-U-RBA-----0	EXTENT-TYPE-----X'00'
VOLFLAG-----PRIME	<b>TRACKS/CA-----5</b>
<b>STRIPE-NUMBER-----3</b>	
EXTENTS:	
LOW-CCHH-----X'02AF0004'	LOW-RBA-----0
<b>TRACKS-----50</b>	
HIGH-CCHH-----X'02B20008'	HIGH-RBA-----7372799

---

There are key differences between the original non-striped data set and the striped one. The striped one has EF/EA tuned on, again only EF is required. EA is included with the Data Class and is convenient for the test. Notice that a Data Class is assigned and so is a different Storage Class. Also notice that the allocation has changed from one extent to three and that cylinders were converted to tracks because of striping. The allocation went from one volume to three volumes.

For this test, the storage group ACS routine has be changed to allocate the new Storage Class of DB9ASDR to the volumes set for Storage Class DB9A. If the storage group ACS routine searches for Storage Classes that are assigned and does not recognize the Storage Class DB9ASDR, you receive the following message in the MSTR STC job log:

```
IGD01013I DATA SET ALLOCATION REQUEST FAILED -
THE ACS STORAGE GROUP ROUTINE DID NOT ASSIGN A STORAGE GROUP
```

The ISMF settings for the Data Class DB2EFEA to enable striping is shown in Figure 4-9.

```
CDS Name      . . . . . : ACTIVE
Data Class Name . . : DB2EFEA

Data Set Name Type . . . . . : EXTENDED
  If Extended . . . . . : REQUIRED
  Extended Addressability . . : YES
  Record Access Bias . . . . : USER
Space Constraint Relief . . . : NO
  Reduce Space Up To (%) . . :
  Dynamic Volume Count . . . :
Compaction . . . . . :
Spanned / Nonspanned . . . . :
Media Interchange
  Media Type . . . . . :
  Recording Technology . . . :
  Performance Scaling . . . . :
  Performance Segmentation . . :
```

Figure 4-9 Data Class DB2EFEA - page 2 of 5. All other settings were default

The ISMF settings for the Storage Class DB9ASDR to enable striping is shown in Figure 4-10.

STORAGE CLASS DISPLAY	Page 1 of 2
Command ==>	
CDS Name . . . . .	: ACTIVE
Storage Class Name	: DB9ASDR
Description	: FOR DB2 APR 2010 RESIDENCY
Performance Objectives	
Direct Millisecond Response . . . .	:
Direct Bias . . . . .	:
Sequential Millisecond Response . .	:
Sequential Bias . . . . .	:
Initial Access Response Seconds . .	:
Sustained Data Rate (MB/sec) . . .	: 12
OAM Sublevel . . . . .	:
Availability . . . . .	: NOPREF
Accessibility . . . . .	: NOPREF
Backup . . . . .	:
Versioning . . . . .	:
STORAGE CLASS DISPLAY	Page 2 of 2
Command ==>	
CDS Name . . . . .	: ACTIVE
Storage Class Name	: DB9ASDR
Guaranteed Space . . . . .	: NO
Guaranteed Synchronous Write . .	: NO
Multi-Tiered SGs . . . . .	:
Parallel Access Volume Capability	: NOPREF
Cache Set Name . . . . .	:
CF Direct Weight . . . . .	:
CF Sequential Weight . . . . .	:
Lock Set Name . . . . .	:

Figure 4-10 Storage Class DB9ASDR

SMS striping differs from stripes attained by RAID technology. The example refers to VSAM striping. Sequential striping is also possible and is something for you to consider.

Stripe all possible objects that are input or output for utilities when the associated table space is striped. For example (when the associated table space is striped), stripe the following items:

- ▶ Image copy data sets
- ▶ Input files for LOADs, etc.

Only disk data sets can be striped.

You can now stripe your disk archive log data sets beginning in DB2 9. The reason is because the archive log data sets were converted from BDAM to BSAM and are now eligible to stripe.



For differences between VSAM striping and sequential striping, see the following resources:

- ▶ *DFSMS Using Data Sets, SC26-7410*
- ▶ *DFSMSdfp Storage Administration, SC26-7402*

From an SMS ISMF and ACS perspective, VSAM or sequential striping uses the same mechanism. The Data Class must still specify EF and the Storage Class must have an SDR rate. As an example, we create a striped sequential data set using the same classes as before: DB2EFEA for the Data Class and DB9ASDR for the Storage Class.

Example 4-5 shows a typical image copy data set that is not striped.

*Example 4-5 Image copy with a non-striped sequential data set*

---

```
//UTIL EXEC DSNUPROC,SYSTEM=DB9A,UID='TEMP'
//SYSCOPYX DD DSN=DB9AU.DSNDBD.JOHNIC.NONSTRIP,
//           DISP=(NEW,CATLG,DELETE),
//           SPACE=(CYL,(510,100),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
COPY TABLESPACE DSNDB04.JOHNITS1 COPYDDN SYSCOPYX
```

The job runs successfully, from an ISPF 3.4 point of view for the image copy data set we find:

Data Set Name . . . . : DB9AU.DSNDBD.JOHNIC.NONSTRIP

General Data	Current Allocation
Management class . . : MCDB22	<b>Allocated cylinders : 330</b>
<b>Storage class . . . : DB9A</b>	<b>Allocated extents . : 1</b>
Volume serial . . . : SB0X1V	
Device type . . . . : 3390	
<b>Data class . . . . . : **None**</b>	
Organization . . . . : PS	Current Utilization
Record format . . . : FB	<b>Used cylinders . . : 330</b>
Record length . . . : 4096	<b>Used extents . . . : 1</b>
Block size . . . . : 24576	
1st extent cylinders: 330	Dates
Secondary cylinders : 100	Creation date . . . : 2010/05/11
<b>Data set name type :</b>	Referenced date . . : 2010/05/11
SMS Compressible. . : NO	Expiration date . . : ***None***

---

Example 4-6 shows what happens when only a Storage Class is provided, but no other SMS classes. In this case, no Data Class was assigned by the ACS routine, but a Storage Class was specified. Although the Storage Class has an entry for SDR, it is not striped because the Data Class that contains EF enablement was not assigned. The allocation works and the jobs complete with a return code is 0, however the object is not striped.

*Example 4-6 Image copy with a non-striped sequential data set*

---

```
//UTIL EXEC DSNUPROC,SYSTEM=DB9A,UID='TEMP'
//SYSCOPYX DD DSN=DB9AU.DSNDBD.JOHNIC.STRIPED,
//           DISP=(NEW,CATLG,DELETE),STORCLAS=DB9ASDR,
//           SPACE=(CYL,(510,100),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
```

```
//SYSIN DD *
COPY TABLESPACE DSNDB04.JOHNITS1 COPYDDN SYSCOPYX
```

The job runs successfully - however the data set is *not* striped, from an ISPF 3.4 point of view for the image copy data set we find:

Data Set Name . . . . : DB9AU.DSNDBD.JOHNIC.STRIPED

General Data	Current Allocation
Management class . . : MCDB22	<b>Allocated cylinders : 330</b>
<b>Storage class . . . : DB9ASDR</b>	<b>Allocated extents . : 1</b>
Volume serial . . . : SBOX0F	
Device type . . . . : 3390	
<b>Data class . . . . . : **None**</b>	
Organization . . . . : PS	Current Utilization
Record format . . . . : FB	<b>Used cylinders . . : 330</b>
Record length . . . . : 4096	<b>Used extents . . . : 1</b>
Block size . . . . . : 24576	
1st extent cylinders: 330	
Secondary cylinders : 100	Dates
<b>Data set name type :</b>	Creation date . . . : 2010/05/11
SMS Compressible. . : NO	Referenced date . . : 2010/05/11
	Expiration date . . : ***None***

---

Example 4-7 shows that the proper Data Class and Storage Class are assigned for both EF and SDR, and the data set is now striped, as in the VSAM striped example, three stripes.

*Example 4-7 Image copy with a striped sequential data set*

---

```
//UTIL EXEC DSNUPROC,SYSTEM=DB9A,UID='TEMP'
//SYSCOPYX DD DSN=DB9AU.DSNDBD.JOHNIC.STRIPED2,
//           DISP=(NEW,CATLG,DELETE),DATACLAS=DB2EFEA,
//           STORCLAS=DB9ASDR,
//           SPACE=(CYL,(510,100),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
COPY TABLESPACE DSNDB04.JOHNITS1 COPYDDN SYSCOPYX
```

The job runs successfully and the data set is now striped, from an ISPF 3.4 point of view for the image copy data set we find:

Data Set Name . . . . : DB9AU.DSNDBD.JOHNIC.STRIPED2

General Data	Current Allocation
Management class . . : MCDB22	<b>Allocated cylinders : 330</b>
<b>Storage class . . . : DB9ASDR</b>	<b>Allocated extents . : 3</b>
<b>Volume serial . . . : SBOX0G +</b>	
Device type . . . . : 3390	
<b>Data class . . . . . : DB2EFEA</b>	
Organization . . . . : PS	Current Utilization
Record format . . . . : FB	<b>Used cylinders . . : 330</b>
Record length . . . . : 4096	<b>Used extents . . . : 3</b>
Block size . . . . . : 24576	
<b>1st extent cylinders: 110</b>	
Secondary cylinders : 100	Dates
<b>Data set name type : EXTENDED</b>	Creation date . . . : 2010/05/11
SMS Compressible. . : NO	Referenced date . . : 2010/05/11

Expiration date . . : \*\*\*None\*\*\*

```
Esaaaaaaaa Volume Information sssssssssN
e
e Command ==> e
e
e All allocated volumes: e
e More: + e
e Number of volumes allocated: 3 e
e
e SB0X0G SB0X0F SB0X0U e
e
e
e
e
e
e
DssssssssssssssssssssssssssssssssssssM
```

---

From a performance perspective, VSAM or sequential striping may be beneficial and must be tested. One key factor to test is the number of stripes required and which data sets would benefit most from striping. Some customers do not stripe their VSAM data sets, but find performance improvements when striping their sequential data sets for DB2 utilities.

#### 4.3.4 SMS management class

The SMS Management Class can be useful for DB2 environments in several ways, for the following environments:

► Production environment

Many users allocate the archive log data sets to disk, DFSMSHsm can migrate the data sets to ML1 which is DFSMSHsm owned disk for quick recall for access, or more commonly for large archive log data sets directly to ML2, which is typically tape. If the data set has been migrated to ML1, then recalls are generally fast because the data set is moved from DFSMSHsm owned disk to the original storage group that houses the archive log data sets. Recall from ML2 brings the data set back to disk. This process works well because, after the object is on disk, multiple RECOVERY jobs can be executed concurrently against the same archive log data set. When archiving to tape, even VTS, which front ends tapes with disk, the device type passed back to DB2 is tape. This means that requests against the archive log data sets will be serialized, elongating recoveries and leading to deadlocks between restores in a data sharing environment. Physical tapes can be extremely large, if recalls are executed for your archive log, and other data sets on the same tape, recalls are serialized because we are not dealing with a direct access device.

Note the following information:

- The storage group that is housing the archive log data sets must be large enough to house the required archive log data sets before being migrated, and any that need to be recalled for recovery.
- Recalls from DFSMSHsm is an automated process, but can be manually executed ahead of time also.
- If both copies of the archive log reside on ML2 tape, be careful to avoid a single point of failure, such as the tape snapping. Alternatives include DFSMSHsm duplexing tape, having another tape system backing up the original set, such as the grid system used by the TS7700, or transmitting a copy offsite.

Image copy data sets follow the same convention as the archive logs outlined previously. Remember single points of failure and serialization if the object resides on the same tape.

Aggregate backup and recovery support (ABARS) can be used to copy data residing on tape or disk.

- ▶ **Non-production environment**

The same items apply to non-production as to production, although certain users might choose to migrate data sets also. If so, verify that ZPARM values for RECALL and RECALLD are set appropriately to avoid DB2 process failures. Migrating DB2 data requires other considerations also. For instance, are objects expired after a certain period? In case a copy of the DB2 subsystem or the Data Sharing group needs to be cloned, migrated items must also be factored in.

Many users find that running the partial release function of objects that are in EF without Guaranteed Space can save a large amount of space. The trade off is slight effect on performance for allocation of the next extent.

For the production environment, most users do not migrate or back up their DB2 LDSs using DFSMSHsm. They can assign the LDSs with a Management Class of MCNOACT, which is a Management Class that takes no action, no migration, and no backups.

To display a Migration Class, use ISMF and enter option 3 for Management Class, then use LIST or DISPLAY. MCNOACT shows that there is no request for DFSMSHsm to process data with this Management Class.

### **4.3.5 SMS storage groups**

The SMS classes provide input for data set allocation assigned by the storage group.

DB2 environments typically have several storage groups based on requirements and production versus non-production designations. Production environments typically have one storage group, based on the following configuration:

- ▶ DB2 catalog and directory
- ▶ Active log and boot strap data sets
- ▶ User table spaces and indexes
- ▶ Work files (DSNDB07 and equivalent), which can reside in their own storage group. An alternative, if a sufficient amount of addresses are assigned using PAV, is to combine the work files in the same storage group as those used by the DB2 catalog and directory or the user table spaces and indexes.
- ▶ Archive log data sets if not created on tape
- ▶ Image copy data sets if not created on tape

Users might choose to segregate the user table spaces and indexes. However, with proper use of PAV the separation is no longer necessary.

The configuration is based on performance, and backup and recovery strategies. There are times when only user table spaces and indexes require a full volume restore or Flash back. The separation allows for flexibility when dealing with performance or volume backup and restore issues.

Use special considerations for the ICF catalogs and aliases relating to these data sets. For example, if all of the DB2 data sets, including image copies use the same alias or a different alias, but in the same ICF catalog, and the catalog resides on a volume that was flashed

back, image copy entries in the ICF catalog are lost from the time of the FlashCopy. Part of a recovery strategy requires analysis of the ICF catalogs and their aliases and placement on volumes based on flash backs or loss of a volume.

Users can combine data sets from the previous categories into the same storage group. If a volume is lost, all of DB2 (and not potentially only a part of it) is not available, because a volume can house data sets for the DB2 catalog and directory, boot strap, and user table spaces and indexes. Recovery can be elongated by trying to recover multiple objects with unlike characteristics instead of concentrating on recovering one type of data. Recovery of an active log data set differs from recovery or rebuilding of an index. Performance is another factor when data is intermixed.

Many users expand on the use of the user table spaces and indexes. Instead of one large storage group, users can provide two (large and small) or three (large, medium, and small) storage groups. The ACS routine has an entry that specifies the size, above or below, or in between to allocate the data set onto a specific storage group based on similar size requirements. The actual size in the ACS routine is user-dependent. Many users have 90% of their data below 300 cylinders and 10% above, therefore there is a clear split regarding size. Use of *sliding secondary* can cause some of the allocations to be allocated to a pool not intended for its use. Unless a large PRIQTY size is specified, non-LOB data sets start allocations off with one cylinder and LOB data sets with 10 cylinders. Increase of space for extents is based on PRIQTY or SECQTY, or both, and the DSSIZE specified.

The problem with size relates to small data sets that occupy space and also may cause some disk fragmentation, and failures when trying to allocate large data sets, especially when a large amount of space is requested and additional space is required during such operations as REORG. Data sets that are large in nature can avoid the small performance penalty of sliding secondary by allocating the data set with a large PRIQTY or SECQTY, or both. Online REORG operations can suffer when trying to allocate the shadow data set when more space in the storage group is necessary during the process and other, especially small data sets use space on all or most volumes.

Large storage groups require a small amount of additional volumes to assure that large data sets have room to expand and for the creation of shadow data sets for utilities.

Segregation also means between Data Sharing groups and subsystems.

If DFSMSHsm is used to migrate and recall any data sets, storage groups must be sized large enough to hold a specified amount of storage and the recalled data.

Non-production environments are user-dependent and the same issues apply as for production. Space and backup/recovery issues must be addressed. Some user can view non-production data with the same importance as production and therefore apply the same rules and categories. Other users combine all of their non-production data onto the same set of volumes thereby bypassing performance and backup/recovery requirements.

Storage group is not an option when in ISMF user mode, you must be in storage administrator (SA) mode. When in SA mode, unlike the classes, there is no DISPLAY option available. The columns can be viewed by paging right or left. For the DISPLAY format, you can enter ALTER as a line-operator command next to the storage group you want to display, or use option 3, Alter. For ALTER, you must provide a proper CDS in the ISMF storage group panel under CDS Name, you cannot use ACTIVE.

The environment, category, and type of an emulated disk determine what the HIGH and LOW values in the storage group should be set to, and the migration requirements. The HIGH and LOW values for a storage group that is housing DB2 user table spaces and indexes, where data sets are not migrated, are typically not the same as the storage group for archive logs

that frequently have their data migrated. Emulated disk with larger sizes can afford to have a larger HIGH value: 85% of a mod 3 has a much smaller amount of space than 85% of a mod 9 with 54 GB.

The break point value (BPV) for EAV volumes is set in the SMS storage group. This value is the determining factor for allocating to the cylinder managed area versus the track-managed area.

### 4.3.6 DB2 STOGROUPs and SMS storage groups

The concepts of DB2 STOGROUP and SMS storage group are different, but similar. Although a storage group refers to a set of volumes in an installation, the STOGROUP refers to a set of volumes containing a set of data. Separate STOGROUPs can share the same disk volume or volumes; a volume can belong to only one storage group.

DB2 administrators sometimes define many STOGROUPs for their applications. Sometimes they have STOGROUPs for each individual volume and use it to direct the table spaces to that specific volume. Other installations have STOGROUPs at database or application level.

To make the best use of DFSMS, DB2 administrators can define their STOGROUPs as before, but using a generic volume reference (VOLUMES '\*'). The following example has a VOLUMES entry, but no SMS classes:

```
CREATE STOGROUP JOHNEFEA VOLUMES('*')
VCAT DB9AU;
```

The generic volume reference allows DFSMS to choose a volume based on the SMS classes that are assigned to a table or index space. Do not use more than one asterisk ('\*') because doing so can limit the number of volumes for allocation. DB2 9 allows STOGROUPs to be defined with SMS class names and no longer requires the VOLUMES parameter.

DB2 storage groups can still be assigned with specific volsers. If the assigned Storage Class does not have Guaranteed Space enabled, a specific requested volume (or volumes) is ignored and the allocation will work as if the asterisk ('\*') was specified. If Guaranteed Space was specified for the object, the request for the volumes in the STOGROUP are honored and only those volumes specified in the STOGROUP are eligible for allocation.

Table 4-1 summarizes these differences.

*Table 4-1 Differences between DB2 STOGROUP and SMS storage group*

DB2 STOGROUP	SMS storage group
Different STOGROUPs can share the same disk volume (or volumes)	One disk volume can belong only to one SMS storage group.
VOLSERs are specific	Specify the SMS constructs in the DB2 CREATE STOGROUP and ignore the VOLUME attribute. Avoid Guaranteed Space and specific VOLSERs where possible because it defeats the purpose of SMS.
SYSIBM.SYSVOLUMES has a row for each volume in column VOLID	When created with VOLUMES('*'), SYSIBM.SYSVOLUMES has an asterisk (*) for column VOLID.
Limited to management of 133 volumes	No volume limit
Volume selection is based on free space.	Volume selection is based on SMS algorithms.

### 4.3.7 Assigning SMS classes and storage groups for DB2 objects

SMS can be used to allocate all DB2 objects that reside on disk, including the DB2 active log and boot strap data sets, sort data sets (DSNDB07 or equivalent), user table space and indexes, archive log data sets, and image copy data sets.

SMS allocates the data sets based on the combination of ISMF entries and ACS routines for the classes and storage group. ACS routines determine which entries in ISMF will actually be used. The Data Class ACS routine works with the data entered into the ISMF Data Class panels, the Storage Class ACS routine works with the data entered into the Storage Class ACS routine panels, the Management Class ACS routine works with the data entered into the Management Class ACS routine panels, and the storage group ACS routine works with the data entered into the storage group ACS routine panels.

After the Data Class ACS routines, the other ACS routines can read settings from any of the previous routines. For example, The Data Class routine looks for a pattern of your data set name and sets a value for an entry called DB2EFEA. The Storage Class does not have to look for your data set name pattern again, it can simply look and see if Data Class DB2EFEA was passed. If DB2EFEA was passed, your Storage Class can now set the value for your Storage Class to be a valid class. The ACS routines can also search a combination of entries, for example, the Storage Class can be coded also. If the Data Class="DB2EFEA" and the data set pattern was equal to yours, set the Storage Class to a specific class. The classes, including the Data Class can determine what actions are taken if a user requests a specific class.

The ACS routines can test against many variables. For example, if the Data Class assigns a specific class or Storage group for all production DB2 table spaces and indexes with a high-level qualifier of DB2PROD, the ACS routines can simply target only data sets with the following pattern:

```
DB2PROD.DSNDB%.**
```

Specific qualifiers can also be targeted, for example we can investigate whether the third qualifier equals DSNDB01 or DSNDB06. Use of wild cards are valid. Many other variables can be analyzed (to name several):

- ▶ Data set name or specific qualifiers
- ▶ Job name
- ▶ UNIT name or number
- ▶ Program name
- ▶ DD name

When the ACS routines are changed and before they are put into place, thoroughly test all routines. ISMF provides a test facility that can test one or a combination of variables to determine the outcome. This testing is done through ISMF panel, option 7 - Automatic Class Selection. Option 4 allows for a test. For example, a test was executed as follows:

```
DSN.DB9AU.DSNDBD.DSNDB04.JOHNITS3.I0001.A001
```

The results are shown in Figure 4-11 on page 234.

The output shows that no Data Class was assigned, the Storage Class was DB9A, the Management Class is MCDB22, and the storage group was DB9A. Based on the test, we must determine whether all the classes and the storage group are set correctly. The Data Class in this example is null; we know that none was set. Was there supposed to be a Data Class set? Testing must always be run before ACS changes are activated.

```

ACS TESTING RESULTS

CDS NAME           : SYS1.SMS.SCDS
ACS ROUTINE TYPES: DC SC MC SG
ACS TEST LIBRARY  : DB2R2.JCL

  ACS TEST
  MEMBER          EXIT CODE   RESULTS
  -----
DESCRIPTION:
EXPECTED RESULT:
ACSTESTD          0  DC = NULL VALUE ASSIGNED
                  0  SC = DB9A
                  0  MC = MCDB22
MSG : MC ACS GETS CONTROL &ACSENVIR=ALLOC
                  0  SG = DB9A

ACS TESTING RC:   00

```

Figure 4-11 ACS test output example

Figure 4-12 illustrates what an ACS routine looks like when various qualifiers are involved. The storage administrator can use various qualifiers to assign classes or storage groups.

```

/*****
/* PARTITION FILTER
*****/

    FILTLIST &PTSP INCLUDE ('LINEITEM','ORDER','PART','PARTSUPP',
                          'SUPPLIER','NATION','REGION')
                          /* Supply a list of the partitioned double-spaces
*/

    FILTLIST &PNDX INCLUDE ('PXL@OK','PXO@OK','PXP@PK','PXPS@SK',
                          'PXS@SK','PXN@NK','PXR@RK')
                          /* Supply a list of the partitioned indexes
*/

    WHEN ( (&DSN(4) = &PTSP OR &DSN(4) = &PNDX)
          AND (&LLQ EQ 'A001' OR &LLQ EQ 'A002') )
      SET &STOGROUP EQ 'SGDB2GRA'

    WHEN ( (&DSN(4) = &PTSP OR &DSN(4) = &PNDX)
          AND (&LLQ EQ 'A003' OR &LLQ EQ 'A004') )
      SET &STOGROUP EQ 'SGDB2GRB'

    WHEN ( (&DSN(4) = &PTSP OR &DSN(4) = &PNDX)
          AND (&LLQ EQ 'A005' OR &LLQ EQ 'A006') )
      SET &STOGROUP EQ 'SGDB2GRC'

/* Repeat the previous WHEN statement for as many STOGROUPs as reqd

```

Figure 4-12 ACS routine extract using table and index name filter list



### 4.3.8 SMS base configuration

The SMS base configuration provides entries for the following items:

- ▶ Default Management Class
- ▶ Default Device Geometry
- ▶ Default Unit
  - Bytes/track
  - Tracks/cylinder
- ▶ Data set name for the Separation Profile
- ▶ LPAR names to which this configuration should be attached

### 4.3.9 Separation profile

Data set separation allows you to designate groups of data sets in which all SMS-managed data sets within a group are kept separate, on the physical control unit (PCU) level or the volume level, from all the other data sets in the same group.

For a DB2 environment, the Separation Profile can be used for various reasons:

- ▶ To separate the active log and boot strap data sets behind different controllers for availability.
- ▶ To separate specific data sets onto different volumes and extent pools during allocation that must not reside on the same volume because they cause hot spots.

Allocations can still work with the classes and the storage group, and with the Separation Profile too. For example, if you know that partitions 1 and 2 must not reside on the same volume because of contention problems causing hot spots, the normal classes and storage group are invoked. However, the entry in the Separation Profile is invoked to provide the final volume allocation.





## Implementing SMS for a DB2 environment

Certain DB2 data sets are standard sequential files or partitioned data sets. Many installations already manage these data sets with SMS and have already SMS classes defined for these data sets. In this chapter, we analyze in detail an SMS-managed DB2 environment.

We describe attributes for SMS management of the DB2 catalog, log and archive data sets, image copy data sets and provide sample SMS constructs for all these data sets. We also include considerations for converting to SMS-managed DB2 data sets.

This chapter contains the following topics:

- ▶ Considerations before implementing SMS
- ▶ Implementing SMS constructs
  - Data Class implementation
  - Storage Class implementation
  - Management Class implementation
  - Storage group implementation
  - Separation Profile
- ▶ Considerations for DB2 data set types
- ▶ Converting from non-SMS to SMS

## 5.1 Considerations before implementing SMS

Before implementing SMS for a DB2 environment, we need to understand what the DB2 requirements are and what hardware and software is available to meet those requirements.

We can then start implementing the SMS constructs, which are described in 4.2, “Storage management with DFSMS” on page 203 for managing the DB2 data sets.

### 5.1.1 DB2 data set considerations

The types of data sets to consider in a DB2 environment are as follows:

- ▶ Boot strap data sets
- ▶ Active log data sets
- ▶ Archive log data sets
- ▶ DB2 catalog and directory data sets
- ▶ DB2 work data sets (DSNDB07 or equivalent)
- ▶ DB2 user table spaces and indexes
- ▶ Image copy data sets
- ▶ Data sets required for different utilities - both DB2 and temporary
- ▶ PDSEs and environmental data sets

### 5.1.2 Environment considerations

Consider what is needed in your environment:

- ▶ What disk is available?
  - Which vendor have I bought my disk from?
  - How many disk boxes are available?
  - How much space is available and how much space can I use for my DB2 environment?
  - How much cache is available?
  - What model disk do I emulate?
  - Is FICON or ESCON used, and what is the bandwidth?
  - What type of RAID is implemented?
  - What type of PAV is implemented?
  - Is FlashCopy installed and if so, what version?
- ▶ Is any VSAM or sequential striping required?
- ▶ Is EF, EA, or both required for any or all data sets?
- ▶ Should Volume Count, Dynamic Volume Count, or both be used, and if so for what purpose?
- ▶ Should we bypass the 255 extent limit for data sets?
- ▶ Should we bypass the 5 extent rule for allocations?
- ▶ Are large sequential format data sets required?
- ▶ Are any entries required in the Separation Profile, and if so, why?
- ▶ Do any data sets require Guaranteed Space, and if so, why?
- ▶ Which data sets should be allocated to separate physical volumes, controllers, or both?

- ▶ Are any data sets migrated, such as table space, indexes, archive logs, image copies? If so, based on what criteria?
- ▶ Should release of unused space be used for any data sets?
- ▶ Should any volumes be backed up using DFSMSHsm or DFSMSdss?
- ▶ How many storage groups should be created and for what use?
- ▶ What should the storage group value for HIGH and LOW be set to for each storage group?
- ▶ Should the Extend Storage Groups, overflow volumes, or both be used?
- ▶ Should multitiered storage groups be used?
- ▶ Are tapes required, and if so what types of tapes are used?

## 5.2 Implementing SMS constructs

This section has a sample implementation of the SMS constructs showing the sequence of the most relevant screens for the following topics:

- ▶ Data Class implementation
- ▶ Storage Class implementation
- ▶ Management Class implementation
- ▶ Storage group implementation
- ▶ Separation Profile

### 5.2.1 Data Class implementation

Many customers do not assign any Data Class for their DB2 data sets. Storage administrators can set up many various Data Classes based on user requirements. Several of the most common reasons to use a Data Class in a DB2 environment are as follows and are referred to (by number) in the subsequent examples:

1. Enabling EF, EA, or both
2. Bypassing the 255 extent rule for data sets
3. Bypassing the 5-extent rule for allocations of data sets
4. Reducing space requirements when no volume meets the space requirement
5. VSAM, sequential striping, or both. Striping requires the Storage Class use of SDR also.
6. Allocating data sets with common DCB, space characteristics, or both
7. Specifying additional volumes for DB2 or utility data sets
8. Specifying different data sets types, such as PDSE, large format, and so on

All these requirements can result in the storage administrator creating one Data Class per requirement or fewer Data Classes combining requirements.

Several of the following examples are combined for the various requirements. The first five requirements are combined into one Data Class called DB2EFEAX. This Data Class can be used for all DB2 user table spaces and indexes.

The ISMF section for the Data Class has five *pages* or panels. All five pages are illustrated, but for the first five requirements that make up DB2EFEAX Data Class, we require only pages 2 and 5. These pages, as illustrated, are from an ISMF DEFINE or ALTER perspective, not DISPLAY, although DISPLAY has similar pages and information.

Data Class DB2EFEAX already exists so option 4, **Alter**, is selected instead of option 3, Define, as shown in Figure 5-1.

```

DATA CLASS APPLICATION SELECTION
Command ==>

To perform Data Class Operations, Specify:
  CDS Name . . . . . 'SYS1.SMS.SCDS'
                                     (1 to 44 character data set name or 'Active' )
  Data Class Name . . DB2EFEAX (For Data Class List, fully or partially
                                     specified or * for all)

Select one of the following options :
  4 1. List   - Generate a list of Data Classes
    2. Display - Display a Data Class
    3. Define  - Define a Data Class
    4. Alter  - Alter a Data Class

If List Option is chosen,
  Enter "/" to select option      Respecify View Criteria
                                   Respecify Sort Criteria

```

Figure 5-1 ISMF panel for Data Class

As Figure 5-2 shows, the page 1 panel does not require any overrides for the first five requirements and it is left at default values.

```

DATA CLASS ALTER                               Page 1 of 5
Command ==>

SCDS Name . . . : SYS1.SMS.SCDS
Data Class Name : DB2EFEAX

To ALTER Data Class, Specify:
  Description ==> First 5 requirements
               ==>
  Recfm . . . . . (any valid RECFM combination or blank)
  Lrecl . . . . . (1 to 32761 or blank)
  Override Space . . . . . N (Y or N)
  Space Avgrec . . . . . (U, K, M or blank)
    Avg Value . . . . . (0 to 65535 or blank)
    Primary . . . . . (0 to 999999 or blank)
    Secondary . . . . . (0 to 999999 or blank)
    Directory . . . . . (0 to 999999 or blank)
  Retpd or Expdt . . . . . (0 to 9999, YYYY/MM/DD or blank)
  Volume Count . . . . . 1 (1 to 255 or blank)
  Add'l Volume Amount . . . (P=Primary, S=Secondary or blank)

```

Figure 5-2 Data Class page 1 of 5: Alter DB2EFEAX

Page 2 fulfills four of the five requirements:

- The data set type is set to extended (EF).
- Extended Addressability (EA) is enabled
- Space Constraint Relief went from a default of N to Y(es)
- Data sets that do not have enough space on any volume have their request reduced by 10% and the request is redriven at the lower amount.

By setting Space Constraint Relief to **YES** and having a space-reduced value, even 0 bypasses the 5-extent rule for allocations. See Figure 5-3.

DATA CLASS ALTER Command ==>  SCDS Name . . . : SYS1.SMS.SCDS Data Class Name : DB2EFEAX  To ALTER Data Class, Specify:  <table style="width: 100%;"> <tr> <td style="width: 40%;"><b>Data Set Name Type</b> . . . . .</td> <td>EXT</td> <td>(EXT, HFS, LIB, PDS, Large or blank)</td> </tr> <tr> <td><b>If Ext</b> . . . . .</td> <td>R</td> <td>(P=Preferred, R=Required or blank)</td> </tr> <tr> <td><b>Extended Addressability</b> . .</td> <td>Y</td> <td>(Y or N)</td> </tr> <tr> <td>Record Access Bias . . . . .</td> <td>U</td> <td>(S=System, U=User or blank)</td> </tr> <tr> <td><b>Space Constraint Relief</b> . . .</td> <td>Y</td> <td>(Y or N)</td> </tr> <tr> <td><b>Reduce Space Up To (%)</b> . .</td> <td>10</td> <td>(0 to 99 or blank)</td> </tr> <tr> <td>Dynamic Volume Count . . . .</td> <td></td> <td>(1 to 59 or blank)</td> </tr> <tr> <td>Compaction . . . . .</td> <td></td> <td>(Y, N, T, G or blank)</td> </tr> <tr> <td>Spanned / Nonspanned . . . . .</td> <td></td> <td>(S=Spanned, N=Nonspanned or blank)</td> </tr> <tr> <td>System Managed Buffering . .</td> <td></td> <td>(1K to 2048M or blank)</td> </tr> <tr> <td>System Determined Blocksize</td> <td>N</td> <td>(Y or N)</td> </tr> <tr> <td>EATTR . . . . .</td> <td></td> <td>(0=Opt, N=No or blank)</td> </tr> </table>	<b>Data Set Name Type</b> . . . . .	EXT	(EXT, HFS, LIB, PDS, Large or blank)	<b>If Ext</b> . . . . .	R	(P=Preferred, R=Required or blank)	<b>Extended Addressability</b> . .	Y	(Y or N)	Record Access Bias . . . . .	U	(S=System, U=User or blank)	<b>Space Constraint Relief</b> . . .	Y	(Y or N)	<b>Reduce Space Up To (%)</b> . .	10	(0 to 99 or blank)	Dynamic Volume Count . . . .		(1 to 59 or blank)	Compaction . . . . .		(Y, N, T, G or blank)	Spanned / Nonspanned . . . . .		(S=Spanned, N=Nonspanned or blank)	System Managed Buffering . .		(1K to 2048M or blank)	System Determined Blocksize	N	(Y or N)	EATTR . . . . .		(0=Opt, N=No or blank)	Page 2 of 5
<b>Data Set Name Type</b> . . . . .	EXT	(EXT, HFS, LIB, PDS, Large or blank)																																			
<b>If Ext</b> . . . . .	R	(P=Preferred, R=Required or blank)																																			
<b>Extended Addressability</b> . .	Y	(Y or N)																																			
Record Access Bias . . . . .	U	(S=System, U=User or blank)																																			
<b>Space Constraint Relief</b> . . .	Y	(Y or N)																																			
<b>Reduce Space Up To (%)</b> . .	10	(0 to 99 or blank)																																			
Dynamic Volume Count . . . .		(1 to 59 or blank)																																			
Compaction . . . . .		(Y, N, T, G or blank)																																			
Spanned / Nonspanned . . . . .		(S=Spanned, N=Nonspanned or blank)																																			
System Managed Buffering . .		(1K to 2048M or blank)																																			
System Determined Blocksize	N	(Y or N)																																			
EATTR . . . . .		(0=Opt, N=No or blank)																																			

Figure 5-3 Data Class page 2 of 5: Alter DB2EFEAX

The panel for page 3 (Figure 5-4 on page 242), does not require any overrides for the first five requirements and is left at default values.

```

DATA CLASS ALTER                               Page 3 of 5
Command ==>

SCDS Name . . . : SYS1.SMS.SCDS
Data Class Name : DB2EFEAX

To ALTER Data Class, Specify:

Media Interchange
Media Type . . . . . (1, 2, 3, 4, 5, 6, 7, 8, 9, 10 or blank)
Recording Technology . . (18,36,128,256,384,E1,E2,EE2,E3,EE3 or ' ')
Performance Scaling . . (Y, N or blank)
Performance Segmentation (Y, N or blank)
Block Size Limit . . . . (32760 to 2GB or blank)
Recorg . . . . . (KS, ES, RR, LS or blank)
Keylen . . . . . (0 to 255 or blank)
Keyoff . . . . . (0 to 32760 or blank)
CIsze Data . . . . . (1 to 32768 or blank)
% Freespace CI . . . . . (0 to 100 or blank)
CA . . . . . (0 to 100 or blank)

```

Figure 5-4 Data Class page 3 of 5: Alter DB2EFEAX

The panel for page 4 (Figure 5-5) does not require any overrides for the first five requirements and is left at default values.

```

DATA CLASS ALTER                               Page 4 of 5
Command ==>

SCDS Name . . . : SYS1.SMS.SCDS
Data Class Name : DB2EFEAX

To ALTER Data Class, Specify:

Encryption Management
Key Label 1 . . . (1 to 64 characters or blank)

Key Label 2 . . .

Encoding for Key Label 1 . . . . . (L, H or blank)
Encoding for Key Label 2 . . . . . (L, H or blank)

```

Figure 5-5 Data Class page 4 of 5: Alter DB2EFEAX

The panel for page 5 (Figure 5-6 on page 243) meets the last requirement, exceeding the 255-extent limit. By default the value is set to N limiting data sets to 255 extents.

Enter **PF3** to save the ALTER changes, and then enable the new SCDS name.



DATA CLASS ALTER		Page 5 of 5
Command ==>		
SCDS Name . . . : SYS1.SMS.SCDS		
Data Class Name : DB2EFEAX		
To ALTER Data Class, Specify:		
Shareoptions Xregion . . .		(1 to 4 or blank)
Xsystem . . .		(3, 4 or blank)
Reuse . . . . . N		(Y or N)
Initial Load . . . . . R		(S=Speed, R=Recovery or blank)
BWO . . . . .		(TC=TYPECICS, TI=TYPEIMS, NO or blank)
Log . . . . .		(N=NONE, U=UNDO, A=ALL or blank)
Logstream Id . . . . .		
FRlog . . . . .		(A=ALL, N=NONE, R=REDO, U=UNDO or blank)
RLS CF Cache Value . . . . A		(A=ALL, N=NONE, U=UPDATESONLY)
RLS Above the 2-GB Bar . . N		(Y or N)
<b>Extent Constraint Removal</b> Y		(Y or N)

Figure 5-6 Data Class page 5 of 5: Alter DB2EFEAX

Data sets assigned with Data Class DB2EFEAX can be listed with LISTCAT and ISPF 3.4. The VSAM cluster name for new data sets assigned display the Data Class name of DB2EFEAX as shown in Example 5-1, output of the LISTCAT.

Example 5-1 LISTCAT for Data Class DB2EFEAX

```

CLUSTER ----- DB9AU.DSNDBC.DSNDB04.JOHNRLSE.I0001.A001
IN-CAT --- UCAT.DB9A
HISTORY
  DATASET-OWNER----PAOLOR9      CREATION-----2010.132
  RELEASE-----2              EXPIRATION-----0000.000
SMSDATA
  STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
  DATACLASS -----DB2EFEAX    LBACKUP ---0000.000.0000
. . .
ASSOCIATIONS
  DATA-----DB9AU.DSNDBD.DSNDB04.JOHNRLSE.I0001.A001
DATA ----- DB9AU.DSNDBD.DSNDB04.JOHNRLSE.I0001.A001
ASSOCIATIONS
  CLUSTER--DB9AU.DSNDBC.DSNDB04.JOHNRLSE.I0001.A001
ATTRIBUTES
SHROPTNS(3,3)    SPEED    UNIQUE    NOERASE    LINEAR    NOWRITECHK
NOIMBED          NOREPLICAT
UNORDERED        REUSE    NONSPANNED    EXTENDED    EXT-ADDR

```

Example 5-1 shows that the data set is assigned a Data Class of DB2EFEAX, and EF and EA are enabled. What you cannot determine from LISTCAT is whether the data set can exceed 255 extents or bypass 5-extents rule. You also cannot determine whether the SCR space was reduced.

As an example for requirement 6, allocating data sets with common DCB, space characteristics (or both), we alter the existing Data Class PE10PO by using the ALTER

option. PE10PO can be used to create a PDS with defaults for the space requirements, and DCB. Only pages 1 and 2 are illustrated, because pages 3 - 5 retain their default values.

Page 1 (Figure 5-7) shows that a combination of DCB and space is specified. For DCB (a JCL parameter), the settings are: RECFM is FB, LRECL is 80, primary allocation is 1 MB, secondary allocation is 1 MB, and number of directory blocks is 10. Volume Count is set to 1 because PDSs and PDSEs cannot exceed one volume.

DATA CLASS ALTER		Page 1 of 5
Command ==>		
SCDS Name . . . : SYS1.SMS.SCDs		
Data Class Name : PE10PO		
To ALTER Data Class, Specify:		
Description ==>		
==>		
<b>Recfm</b> . . . . .	<b>FB</b>	(any valid RECFM combination or blank)
<b>Lrecl</b> . . . . .	<b>80</b>	(1 to 32761 or blank)
Override Space . . . . .	<b>N</b>	(Y or N)
<b>Space Avgrec</b> . . . . .	<b>M</b>	(U, K, M or blank)
Avg Value . . . . .	<b>1</b>	(0 to 65535 or blank)
<b>Primary</b> . . . . .	<b>1</b>	(0 to 999999 or blank)
<b>Secondary</b> . . . . .	<b>1</b>	(0 to 999999 or blank)
<b>Directory</b> . . . . .	<b>10</b>	(0 to 999999 or blank)
Retpd or Expdt . . . . .		(0 to 9999, YYYY/MM/DD or blank)
<b>Volume Count</b> . . . . .	<b>1</b>	(1 to 255 or blank)
Add'l Volume Amount . . . . .		(P=Primary, S=Secondary or blank)

Figure 5-7 Data Class page 1 of 5: Alter PE10PO

On page 2 (Figure 5-8 on page 245), System Determined Blocksize is set to Y, which specifies that the system-determined block size is to be used regardless of the existence of a user-specified block size.

DATA CLASS ALTER	Page 2 of 5
Command ==>	
SCDS Name . . . : SYS1.SMS.SCDS	
Data Class Name : PE10PO	
To ALTER Data Class, Specify:	
Data Set Name Type . . . . .	(EXT, HFS, LIB, PDS, Large or blank)
If Ext . . . . .	(P=Preferred, R=Required or blank)
Extended Addressability . . N	(Y or N)
Record Access Bias . . . . .	(S=System, U=User or blank)
Space Constraint Relief . . . N	(Y or N)
Reduce Space Up To (%) . . .	(0 to 99 or blank)
Dynamic Volume Count . . . .	(1 to 59 or blank)
Compaction . . . . .	(Y, N, T, G or blank)
Spanned / Nonspanned . . . . .	(S=Spanned, N=Nonspanned or blank)
System Managed Buffering . . .	(1K to 2048M or blank)
<b>System Determined Blocksize</b> Y	(Y or N)
EATTR . . . . .	(O=Opt, N=No or blank)

Figure 5-8 Data Class page 2 of 5: Alter PE10PO.

Example 5-2 shows IEFBR14 being executed to invoke Data Class PE10PO. Although an SMS Data Class is used and validated by message IGD100I, data set DB2R2.PDS is not an SMS-managed data set and resides on a volume that is not SMS-managed. The block size is calculated using the system-determined block size.

Example 5-2 IEFBR14 example using Data Class

```
//ALLOC EXEC PGM=IEFBR14
//SYSPRINT DD SYSOUT=*
//DD1 DD DSN=DB2R2.PDS,UNIT=SYSALLDA,DATACLAS=PE10PO,
// DISP=(,CATLG,DELETE)

IEF236I ALLOC. FOR DB2R2B ALLOC
IEF237I JES2 ALLOCATED TO SYSPRINT
IGD100I AC07 ALLOCATED TO DDNAME DD1          DATACLAS (PE10PO)
IEF142I DB2R2B ALLOC - STEP WAS EXECUTED - COND CODE 0000
IEF285I  DB2R2.DB2R2B.JOB05894.D0000101.?          SYSOUT
IEF285I  DB2R2.PDS                                CATALOGED
IEF285I  VOL SER NOS= SBOXEC.

ISPF 3.4 for data set DB2R2.PDS:
Data Set Name . . . . : DB2R2.PDS

General Data                                Current Allocation
Management class . . . : **None**           Allocated megabytes : 1
Storage class . . . . : **None**            Allocated extents . : 1
Volume serial . . . . : SBOXEC
Device type . . . . . : 3390
Data class . . . . . : **None**
Organization . . . . : PO                  Current Utilization
Record format . . . . : FB                 Used megabytes . . . : 1
Record length . . . . : 80                 Used extents . . . . : 1
```

<b>Block size . . . . : 27920</b>	
1st extent megabytes: 1	
Secondary megabytes : 1	Dates
Data set name type : PDS	Creation date . . . : 2010/05/18
	Referenced date . . : ***None***
	Expiration date . . : ***None***

---

Certain parameters can be overridden, but in this case because system determined block size will be used, the block size cannot be overridden.

Example 5-3 shows the IEFBR14 being used again to invoke Data Class PE10P0. This time, the LRECL is being overridden, the Data Class specifies 80, but our JCL specified 50. The JCL also specifies a block size of 500. In this case the LRECL is changed to our request of 50, however the block size is not changed as requested by our JCL because system-determined block size is being used. Note that the job returns a condition code 0.

*Example 5-3 Example of overriding the LRECL and trying to override the block size*

---

```
//ALLOC EXEC PGM=IEFBR14
//SYSPRINT DD SYSOUT=*
//DD1 DD DSN=DB2R2.PDS,UNIT=SYSALLDA,DATACLAS=PE10P0,
// DISP=(,CATLG,DELETE),LRECL=50,BLKSIZE=500

IEF236I ALLOC. FOR DB2R2B ALLOC
IEF237I JES2 ALLOCATED TO SYSPRINT
IGD100I 6D16 ALLOCATED TO DDNAME DD1 DATACLAS (PE10P0)
IEF142I DB2R2B ALLOC - STEP WAS EXECUTED - COND CODE 0000
IEF285I DB2R2.DB2R2B.JOB05908.D0000101.? SYSOUT
IEF285I DB2R2.PDS CATALOGED
IEF285I VOL SER NOS= SBOXFG.
```

ISPF 3.4 for data set DB2R2.PDS:  
Data Set Name . . . . : DB2R2.PDS

General Data	Current Allocation
Management class . . : **None**	<b>Allocated megabytes : 1</b>
Storage class . . . : **None**	<b>Allocated extents . : 1</b>
Volume serial . . . : SBOXFG	
Device type . . . . : 3390	
<b>Data class . . . . : **None**</b>	
<b>Organization . . . : PO</b>	Current Utilization
<b>Record format . . . : FB</b>	Used megabytes . . : 1
<b>Record length . . . : 50</b>	Used extents . . . : 1
<b>Block size . . . . : 27950</b>	
1st extent megabytes: 1	Dates
Secondary megabytes : 1	Creation date . . . : 2010/05/18
Data set name type : PDS	Referenced date . . : ***None***
	Expiration date . . : ***None***

---

The Data Class can be used to allocate other data set types also. For example, we can allocate a VSAM KSDS file, we use Data Class PE10VS to allocate a KSDS.

Page 1 (Figure 5-9 on page 247) shows that the record length is 4400, space is overridden, primary allocation is 1MB, secondary allocation is 1MB. Only one volume is allowed for the allocation.

DATA CLASS ALTER		Page 1 of 5
Command ==>		
SCDS Name . . . : SYS1.SMS.SCDS		
Data Class Name : PE10VS		
To ALTER Data Class, Specify:		
Description ==>		
Recfm . . . . . (any valid RECFM combination or blank)		
Lrecl . . . . . 4400 (1 to 32761 or blank)		
Override Space . . . . . Y (Y or N)		
Space Avgrec . . . . . M (U, K, M or blank)		
Avg Value . . . . . 1 (0 to 65535 or blank)		
Primary . . . . . 1 (0 to 999999 or blank)		
Secondary . . . . . 1 (0 to 999999 or blank)		
Directory . . . . . (0 to 999999 or blank)		
Retpd or Expdt . . . . . (0 to 9999, YYYY/MM/DD or blank)		
Volume Count . . . . . 1 (1 to 255 or blank)		
Add'l Volume Amount . . . . . (P=Primary, S=Secondary or blank)		

Figure 5-9 Data Class PE10VS: Page 1 of 5

Page 2 (Figure 5-10) shows that SDB is enabled.

DATA CLASS ALTER		Page 2 of 5
Command ==>		
SCDS Name . . . : SYS1.SMS.SCDS		
Data Class Name : PE10VS		
To ALTER Data Class, Specify:		
Data Set Name Type . . . . . (EXT, HFS, LIB, PDS, Large or blank)		
If Ext . . . . . (P=Preferred, R=Required or blank)		
Extended Addressability . . N (Y or N)		
Record Access Bias . . . . . (S=System, U=User or blank)		
Space Constraint Relief . . . N (Y or N)		
Reduce Space Up To (%) . . . (0 to 99 or blank)		
Dynamic Volume Count . . . . (1 to 59 or blank)		
Compaction . . . . . (Y, N, T, G or blank)		
Spanned / Nonspanned . . . . . (S=Spanned, N=Nonspanned or blank)		
System Managed Buffering . . . (1K to 2048M or blank)		
System Determined Blocksize Y (Y or N)		
EATTR . . . . . (0=Opt, N=No or blank)		

Figure 5-10 Data Class PE10VS: Page 2 of 5

Page 3 (Figure 5-11 on page 248) shows the allocation is a KSDS file, the key length is 30, offset is 0, CI size is 20480, and the free space for CI and CA are each 0.

```

DATA CLASS ALTER                               Page 3 of 5
Command ==>

SCDS Name . . . : SYS1.SMS.SCDS
Data Class Name : PE10VS

To ALTER Data Class, Specify:

Media Interchange
Media Type . . . . . (1, 2, 3, 4, 5, 6, 7, 8, 9, 10 or blank)
Recording Technology . . (18,36,128,256,384,E1,E2,EE2,E3,EE3 or ' ')
Performance Scaling . . (Y, N or blank)
Performance Segmentation (Y, N or blank)
Block Size Limit . . . . (32760 to 2GB or blank)
Recorg . . . . . KS (KS, ES, RR, LS or blank)
Keylen . . . . . 30 (0 to 255 or blank)
Keyoff . . . . . 0 (0 to 32760 or blank)
CIsze Data . . . . . 20480 (1 to 32768 or blank)
% Freespace CI . . . . . 0 (0 to 100 or blank)
CA . . . . . 0 (0 to 100 or blank)

```

Figure 5-11 Data Class PE10VS: Page 3 of 5

Page 4 (Figure 5-12) is uses the defaults.

```

DATA CLASS ALTER                               Page 4 of 5
Command ==>

SCDS Name . . . : SYS1.SMS.SCDS
Data Class Name : PE10VS

To ALTER Data Class, Specify:

Encryption Management
Key Label 1 . . . (1 to 64 characters or blank)

Key Label 2 . . .

Encoding for Key Label 1 . . . . . (L, H or blank)
Encoding for Key Label 2 . . . . . (L, H or blank)

```

Figure 5-12 Data Class PE10VS - page 4 of 5

Page 5 (Figure 5-13 on page 249) shows share options for XREGION is 2, and XSYSTEM is 3.

DATA CLASS ALTER	Page 5 of 5
Command ==>	
SCDS Name . . . : SYS1.SMS.SCDS	
Data Class Name : PE10VS	
To ALTER Data Class, Specify:	
<b>Shareoptions</b>	
<b>Xregion . . . 2</b>	(1 to 4 or blank)
<b>Xsystem . . . 3</b>	(3, 4 or blank)
Reuse . . . . . N	(Y or N)
Initial Load . . . . . R	(S=Speed, R=Recovery or blank)
BWO . . . . .	(TC=TYPECICS, TI=TYPEIMS, NO or blank)
Log . . . . .	(N=NONE, U=UNDO, A=ALL or blank)
Logstream Id . . . . .	
FRlog . . . . .	(A=ALL, N=NONE, R=REDO, U=UNDO or blank)
RLS CF Cache Value . . . . A	(A=ALL, N=NONE, U=UPDATESONLY)
RLS Above the 2-GB Bar . . N	(Y or N)
Extent Constraint Removal N	(Y or N)

Figure 5-13 Data Class PE10VS: Page 5 of 5

We define cluster by associating it to DATACLASS PE10VS with the JCL in Example 5-4.

Example 5-4 DEFINE using Data Class PE10VS

---

```
//DSNTDBL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DEFINE CLUSTER -
    ( NAME (DB9AU.DSNDBD.TRY.DCKSDS) -
      DATACLASS(PE10VS))

  DEFINE CLUSTER -
    ( NAME (DB9AU.DSNDBD.TRY.DCKSDS) -
      DATACLASS(PE10VS))
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX1Y IS 0
IDC0509I INDEX ALLOCATION STATUS FOR VOLUME SBOX1Y IS 0
IDC0512I NAME GENERATED-(D) DB9AU.DSNDBD.TRY.DCKSDS.DATA
IDC0512I NAME GENERATED-(I) DB9AU.DSNDBD.TRY.DCKSDS.INDEX
IDC0181I STORAGECLASS USED IS DB9A
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0181I DATACLASS USED IS PE10VS
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

---

LISTCAT output in Example 5-5 shows that the default information from the Data Class was used.

**Example 5-5 LISTCAT output after using Data Class PE10VS**

---

```

OCLUSTER ----- DB9AU.DSNDBD.TRY.DCKSDS
  IN-CAT --- UCAT.DB9A
  HISTORY
    DATASET-OWNER----- (NULL)      CREATION-----2010.138
    RELEASE-----2      EXPIRATION-----0000.000
  SMSDATA
    STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDDB2
    DATACLASS -----PE10VS      LBACKUP ---0000.000.0000
    EATTR----- (NULL)
    BWO STATUS-----00000000      BWO TIMESTAMP---00000 00:00:00.0
    BWO----- (NULL)

  ASSOCIATIONS
    DATA-----DB9AU.DSNDBD.TRY.DCKSDS.DATA
    INDEX-----DB9AU.DSNDBD.TRY.DCKSDS.INDEX
0  DATA ----- DB9AU.DSNDBD.TRY.DCKSDS.DATA
    IN-CAT --- UCAT.DB9A

  ATTRIBUTES
    KEYLEN-----30      AVGLRECL-----4400      BUFSPACE-----41984      CISIZE-----20480
    RKP-----0      MAXLRECL-----4400      EXCPEXIT----- (NULL)      CI/CA-----37
    SHROPTNS(2,3)  RECOVERY  UNIQUE      NOERASE      INDEXED      NOWRITECHK      NOIMBED      NOREPLICAT
    UNORDERED      NOREUSE      NONSPANNED

  STATISTICS
    REC-TOTAL-----0      SPLITS-CI-----0      EXCPS-----0
    REC-DELETED-----0      SPLITS-CA-----0      EXTENTS-----1
    REC-INSERTED-----0      FREESPACE-%CI-----0      SYSTEM-TIMESTAMP:
    REC-UPDATED-----0      FREESPACE-%CA-----0      X'0000000000000000'
    REC-RETRIEVED-----0      FREESPC-----1515520

  ALLOCATION
    SPACE-TYPE-----CYLINDER      HI-A-RBA-----1515520
    SPACE-PRI-----2      HI-U-RBA-----0
    SPACE-SEC-----2

0  INDEX ----- DB9AU.DSNDBD.TRY.DCKSDS.INDEX
    IN-CAT --- UCAT.DB9A
    HISTORY
      DATASET-OWNER----- (NULL)      CREATION-----2010.138
      RELEASE-----2      EXPIRATION-----0000.000
      PROTECTION-PSWD----- (NULL)      RACF----- (NO)
    ASSOCIATIONS
      CLUSTER--DB9AU.DSNDBD.TRY.DCKSDS
    ATTRIBUTES
      KEYLEN-----30      AVGLRECL-----0      BUFSPACE-----0      CISIZE-----1024
      RKP-----0      MAXLRECL-----1017      EXCPEXIT----- (NULL)      CI/CA-----33
      SHROPTNS(2,3)  RECOVERY  UNIQUE      NOERASE      NOWRITECHK      NOIMBED      NOREPLICAT      UNORDERED
      NOREUSE
    STATISTICS
      REC-TOTAL-----0      SPLITS-CI-----0      EXCPS-----0      INDEX:
      REC-DELETED-----0      SPLITS-CA-----0      EXTENTS-----1      LEVELS-----0
      REC-INSERTED-----0      FREESPACE-%CI-----0      SYSTEM-TIMESTAMP:      ENTRIES/SECT-----6
      REC-UPDATED-----0      FREESPACE-%CA-----0      X'0000000000000000'      SEQ-SET-RBA-----0
      REC-RETRIEVED-----0      FREESPC-----33792      HI-LEVEL-RBA-----0
    ALLOCATION
      SPACE-TYPE-----TRACK      HI-A-RBA-----33792
      SPACE-PRI-----1      HI-U-RBA-----0
      SPACE-SEC-----1

```

---



The data set is deleted and defined again, this time specifying Data Class PE10VS, and a space parameter as an override. The DEFINE JCL is shown in Example 5-6.

*Example 5-6 DEFINE using Data Class PE10VS with an override of the space parameter*

---

```
//DSNTDBL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
    DEFINE CLUSTER -
        ( NAME (DB9AU.DSNDBD.TRY.DCKSDS) -
          DATACLASS(PE10VS) -
          CYLINDERS(55,25))

    DEFINE CLUSTER -
        ( NAME (DB9AU.DSNDBD.TRY.DCKSDS) -
          DATACLASS(PE10VS) -
          CYLINDERS(55,25))
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOXOF IS 0
IDC0509I INDEX ALLOCATION STATUS FOR VOLUME SBOXOF IS 0
IDC0512I NAME GENERATED-(D) DB9AU.DSNDBD.TRY.DCKSDS.DATA
IDC0512I NAME GENERATED-(I) DB9AU.DSNDBD.TRY.DCKSDS.INDEX
IDC0181I STORAGECLASS USED IS DB9A
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0181I DATACLASS USED IS PE10VS
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

---

Example 5-7 shows the LISTCAT output.

*Example 5-7 LISTCAT output after using Data Class PE10VS with an override of space*

---

```
OCLUSTER ----- DB9AU.DSNDBD.TRY.DCKSDS
IN-CAT --- UCAT.DB9A
HISTORY
  DATASET-OWNER----- (NULL)      CREATION-----2010.138
  RELEASE-----2      EXPIRATION-----0000.000
SMSDATA
  STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
  DATACLASS -----PE10VS      LBACKUP ---0000.000.0000

ASSOCIATIONS
  DATA----DB9AU.DSNDBD.TRY.DCKSDS.DATA
  INDEX----DB9AU.DSNDBD.TRY.DCKSDS.INDEX
0 DATA ----- DB9AU.DSNDBD.TRY.DCKSDS.DATA
IN-CAT --- UCAT.DB9A
HISTORY
  DATASET-OWNER----- (NULL)      CREATION-----2010.138
  RELEASE-----2      EXPIRATION-----0000.000
  ACCOUNT-INFO----- (NULL)
  PROTECTION-PSWD---- (NULL)      RACF----- (NO)
ASSOCIATIONS
  CLUSTER--DB9AU.DSNDBD.TRY.DCKSDS
ATTRIBUTES
  KEYLEN-----30      AVGLRECL-----4400      BUFSPACE-----61440      CISIZE-----20480
  RKP-----0      MAXLRECL-----4400      EXCPEXIT----- (NULL)      CI/CA-----37
  SHROPTNS(2,3)  RECOVERY  UNIQUE      NOERASE      INDEXED      NOWRITECHK      NOIMBED      NOREPLICAT
  UNORDERED      NOREUSE      NONSPANNED
STATISTICS
  REC-TOTAL-----0      SPLITS-CI-----0      EXCPS-----0
```

```

REC-DELETED-----0      SPLITS-CA-----0      EXTENTS-----1
REC-INSERTED-----0     FREESPACE-%CI-----0     SYSTEM-TIMESTAMP:
REC-UPDATED-----0      FREESPACE-%CA-----0      X'0000000000000000'
REC-RETRIEVED-----0    FREESPC-----1515520
ALLOCATION
SPACE-TYPE-----CYLINDER  HI-A-RBA-----1515520
SPACE-PRI-----2        HI-U-RBA-----0
SPACE-SEC-----2

0 INDEX ----- DB9AU.DSNDBD.TRY.DCKSDS.INDEX
IN-CAT --- UCAT.DB9A
HISTORY
DATASET-OWNER----(NULL)   CREATION-----2010.138
RELEASE-----2          EXPIRATION-----0000.000
PROTECTION-PSWD----(NULL) RACF------(NO)
ASSOCIATIONS
CLUSTER--DB9AU.DSNDBD.TRY.DCKSDS
ATTRIBUTES
KEYLEN-----30          AVGLRECL-----0          BUFSPACE-----0          CISIZE-----20480
RKP-----0             MAXLRECL-----20473      EXCPEXIT------(NULL)    CI/CA-----2
SHROPTNS(2,3) RECOVERY  UNIQUE             NOERASE      NOWRITECHK      NOIMBED      NOREPLICAT      UNORDERED
NOREUSE
STATISTICS
REC-TOTAL-----0        SPLITS-CI-----0          EXCPS-----0            INDEX:
REC-DELETED-----0     SPLITS-CA-----0          EXTENTS-----1          LEVELS-----0
REC-INSERTED-----0    FREESPACE-%CI-----0     SYSTEM-TIMESTAMP:        ENTRIES/SECT-----6
REC-UPDATED-----0     FREESPACE-%CA-----0      X'0000000000000000'      SEQ-SET-RBA-----0
REC-RETRIEVED-----0   FREESPC-----81920        HI-LEVEL-RBA-----0
ALLOCATION
SPACE-TYPE-----TRACK    HI-A-RBA-----81920
SPACE-PRI-----2        HI-U-RBA-----0
SPACE-SEC-----2

```

---

Although CYLINDERS(55,25) was specified, the space allocation has not changed from the earlier example. The reason is because the Data Class specified **Override Space - Yes** on Page 1. Even though a different space allocation was requested, the space allocation does not change because override space was set to yes.

Requirement seven is about specifying additional volumes for DB2 or utility data sets. Additional volumes can be used for such things as the number of volumes for *image copies*. Perhaps image copy data sets should have a volume count of 12, which can be set in the Data Class. More often, you work with Volume Count (VC) and Dynamic Volume Count (DVC) for user-managed data sets that are created with an IDCAMS DEFINE, such as the *DB2 catalog and directory data sets*.

When you specify a VC greater than 1, a LISTCAT of a data set using that Data Class will show the VOLSER for the volume on which the data set was allocated; the remaining volumes show them as candidate volumes with VOLSER of '\*' (asterisk in single quotation marks). DVC, however, show only the volume that the data set resides on without the candidate volumes. VC and DVC can be specified together.

Data Class DB2EFEAS was created with VC=3, and DVC=5.

Page 1 (Figure 5-14) controls the Volume Count, which equals 3.

DATA CLASS ALTER		Page 1 of 5
Command ==>		
SCDS Name . . . : SYS1.SMS.SCDS		
Data Class Name : DB2EFEAS		
To ALTER Data Class, Specify:		
Description ==>		
==>		
Recfm . . . . .		(any valid RECFM combination or blank)
Lrecl . . . . .		(1 to 32761 or blank)
Override Space . . . . . N		(Y or N)
Space Avgrec . . . . .		(U, K, M or blank)
Avg Value . . . . .		(0 to 65535 or blank)
Primary . . . . .		(0 to 999999 or blank)
Secondary . . . . .		(0 to 999999 or blank)
Directory . . . . .		(0 to 999999 or blank)
Retpd or Expdt . . . . .		(0 to 9999, YYYY/MM/DD or blank)
<b>Volume Count . . . . . 3</b>		(1 to 255 or blank)
Add'l Volume Amount . . . . .		(P=Primary, S=Secondary or blank)

Figure 5-14 Data Class DB2EFEAS: Page 1 of 5

Page 2 (Figure 5-15) controls the Dynamic Volume Count, which equals 5.

DATA CLASS ALTER		Page 2 of 5
Command ==>		
SCDS Name . . . : SYS1.SMS.SCDS		
Data Class Name : DB2EFEAS		
To ALTER Data Class, Specify:		
Data Set Name Type . . . . . EXT		(EXT, HFS, LIB, PDS, Large or blank)
If Ext . . . . . R		(P=Preferred, R=Required or blank)
Extended Addressability . . . Y		(Y or N)
Record Access Bias . . . . . U		(S=System, U=User or blank)
Space Constraint Relief . . . . Y		(Y or N)
Reduce Space Up To (%) . . . 0		(0 to 99 or blank)
<b>Dynamic Volume Count . . . . 5</b>		(1 to 59 or blank)
Compaction . . . . .		(Y, N, T, G or blank)
Spanned / Nonspanned . . . . .		(S=Spanned, N=Nonspanned or blank)
System Managed Buffering . . . .		(1K to 2048M or blank)
System Determined Blocksize . . N		(Y or N)
EATTR . . . . .		(0=Opt, N=No or blank)

Figure 5-15 Data Class DB2EFEAS: Page 2 of 5

Defining the cluster using the DB2EFEAS data class is shown in Example 5-8.

*Example 5-8 Example of DEFINE using Data Class DB2EFEAS*

---

```
//DSNTIC EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
  DEFINE CLUSTER -
    ( NAME (DB9AU.DSNDBD.TEST.VOLCOUNT) -
      TRACK(1,1) -
      DATACLASS(DB2EFEAS) -
      LINEAR )
  DATA
    ( NAME (DB9AU.DSNDBD.TEST.VOLCOUNT.DATA))
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IGD17070I DATA SET DB9AU.DSNDBD.TEST.VOLCOUNT
ALLOCATED SUCCESSFULLY WITH 1 STRIPE(S).
IGD17172I DATA SET DB9AU.DSNDBD.TEST.VOLCOUNT
IS ELIGIBLE FOR EXTENDED ADDRESSABILITY
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX0G IS 0
IDC0508I DATA ALLOCATION STATUS FOR VOLUME * IS 0
IDC0508I DATA ALLOCATION STATUS FOR VOLUME * IS 0
IDC0181I STORAGECLASS USED IS DB9A
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0181I DATACLASS USED IS DB2EFEAS
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

---

We execute LISTCAT, which shows the VOLSER for the volume that the data set was allocated. See Example 5-9

*Example 5-9 LISTCAT of data set DB9AU.DSNDBD.TEST.VOLCOUNT*

---

```
CLUSTER ----- DB9AU.DSNDBD.TEST.VOLCOUNT
IN-CAT --- UCAT.DB9A
HISTORY
  DATASET-OWNER----- (NULL)      CREATION-----2010.120
  RELEASE-----2      EXPIRATION-----0000.000
SMSDATA
  STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
  DATACLASS -----DB2EFEAS      LBACKUP ---0000.000.0000
. . .
ASSOCIATIONS
  DATA-----DB9AU.DSNDBD.TEST.VOLCOUNT.DATA
DATA ----- DB9AU.DSNDBD.TEST.VOLCOUNT.DATA
IN-CAT --- UCAT.DB9A
. . .
VOLUME
VOLSER-----SBOX0G
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----*
DEVTYPE-----X'3010200F'
```

```

VOLFLAG-----CANDIDATE
VOLUME
VOLSER-----*
DEVTYPE-----X'3010200F'
VOLFLAG-----CANDIDATE

```

---

With the VC set to 3 and a DVC set to 5, after the allocation, we find the data set allocated on one volume with two additional candidate volumes.

Now, the remainder of the volume is blocked with “dummy” space that is provided by sequential data sets with the Storage Class Guaranteed Space attribute. IDCAMS REPRO is executed from one DB2 data set already primed with data, to the test data set, forcing the data set to acquire multi-volume extent.

The data set now resides on two volumes with one as *candidate*. See Figure 5-16.

```

VOLUME
VOLSER-----SBOX0G
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1U
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----*
DEVTYPE-----X'3010200F'
VOLFLAG-----CANDIDATE

```

Figure 5-16 LISTCAT after REPRO

By continuing with more blocking of space and REPROs, the data set now resides on three volumes. See Figure 5-17.

```

VOLUME
VOLSER-----SBOX0G
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1U
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1Z
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME

```

Figure 5-17 LISTCAT after more REPROs

The same process continues. The data set now resides on four volumes. See Figure 5-18.

```

VOLUME
VOLSER-----SBOX0G
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1U
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1Z
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1V
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME

```

Figure 5-18 LISTCAT after more REPROs

All three volumes specified by VC were used and exceeded. We are now at the DVC limit of five volumes. See Figure 5-19.

```

VOLUME
VOLSER-----SBOX0G
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1U
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1Z
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX1V
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME
VOLUME
VOLSER-----SBOX0T
DEVTYPE-----X'3010200F'
VOLFLAG-----PRIME

```

Figure 5-19 LISTCAT after more REPROs

More blocking of space and REPROs continue. After all five volumes are used, when the next REPRO job tried to allocate a new volume, it had reached the DVC threshold of five volumes and abended. See Example 5-10 on page 257.

Example 5-10 Final REPRO

```
IEC070I 209-220,DB2R2$,DSNTDBL,DD2,8416,SBOX0T, 050
IEC070I DB9AU.DSNDBD.TEST.VOLCOUNT,DB9AU.DSNDBD.TEST.VOLCOUNT.DATA,
IEC070I UCAT.DB9A
```

```
REPRO INFILE(DD1) -
      OUTFILE(DD2)
IDC3302I ACTION ERROR ON DB9AU.DSNDBD.TEST.VOLCOUNT
IDC3351I ** VSAM I/O RETURN CODE IS 28 - RPLFDBWD = X'2908001C'
IDC31467I MAXIMUM ERROR LIMIT REACHED.
IDC0005I NUMBER OF RECORDS PROCESSED WAS 2796
IDC3003I FUNCTION TERMINATED. CONDITION CODE IS 12
```

```
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 12
```

DB2 9 allows you to create or alter a DB2 STOGROUP, adding one of the SMS Classes, and no longer requires a VOLUMES parameter. If VOLUMES is omitted, normal DB2 allocation is no longer in effect during EOV processing, rather the SMS Data Class specifications for VC and DVC are in effect. By default, VC is set to 1 and DVC is null. If VOLUMES is omitted and VC, DVC (or both) are not increased, the data set cannot span volumes, and adheres to the request of VC=1.

The last requirement is about different formats of data sets. Only page 2 is illustrated in Figure 5-20 because that is where the data set format is set.

DATA CLASS ALTER	Page 2 of 5
Command ==>	
SCDS Name . . . : SYS1.SMS.SCDS	
Data Class Name : WELCHLRG	
To ALTER Data Class, Specify:	
<b>Data Set Name Type . . . . .</b>	<b>LARGE</b> (EXT, HFS, LIB, PDS, Large or blank)
If Ext . . . . .	(P=Preferred, R=Required or blank)
Extended Addressability . . N	(Y or N)
Record Access Bias . . . . .	(S=System, U=User or blank)
Space Constraint Relief . . . N	(Y or N)
Reduce Space Up To (%) . . .	(0 to 99 or blank)
Dynamic Volume Count . . . .	(1 to 59 or blank)
Compaction . . . . .	(Y, N, T, G or blank)
Spanned / Nonspanned . . . . .	(S=Spanned, N=Nonspanned or blank)
System Managed Buffering . . .	(1K to 2048M or blank)
System Determined Blocksize N	(Y or N)
EATTR . . . . .	(O=Opt, N=No or blank)

Figure 5-20 Data Class WELCGLRG

The types of objects for data set name type are EXTENDED, HFS, LIBRARY, PDS, and LARGE.

In this case, we have a Data Class type of large. For more information regarding this option, see 3.5, “Large sequential data sets” on page 132.

## 5.2.2 Storage Class implementation

In the past, storage administrators had to spend a significant amount of time planning out Storage Class requirements. The requirements involved such things as the speed of storage devices, amount of cache available, whether the disk box allowed for concurrent copy or FlashCopy, and other issues relating to the disk box itself. Generally, in today's technology, most of the Storage Class information is no longer required. The three most common options for a DB2 environment are as follows:

- ▶ Enabling Guaranteed Space
- ▶ Set striping. Striping also requires an associated Data Class with EF enabled.
- ▶ Enabling the use of multitiered storage groups. Multitiered storage groups are specified in the storage group, however you must still enable them in the Storage Class. More information about multitiered storage groups is in 5.2.4, "Storage group implementation" on page 273.

The three most basic Storage Class uses can be combined as in the case of the Data Class, however, they are commonly used separately.

Although we are discussing the ISMF options, the Storage Class is the point when SMS determines whether a data set is SMS-managed. The decision is made in the Storage Class ACS routine. Non-SMS-managed data sets can use certain Data Class functions, then be allocated as non-SMS during the invocation of the Storage Class routines. For example, you can create a non-SMS-managed data set that uses a Data Classes DCB and space information, but other Data Class specifications such as enabling EA, or exceeding the 255-extent rule are reserved for SMS-managed data sets only. See Example 5-11.

*Example 5-11 Storage Class ACS routine specifying some non-SMS-managed data sets*

---

```
PROC STORCLAS
FILTLIST DB9ANOT1 INCLUDE(DB9AU.DSNDB%.*.NONSMS*.**)
  IF &DSN = &DB9ANOT1
  THEN DO
    SET &STORCLAS = ''
  EXIT
END
```

---

Choose ISMF option **5** for the Storage Class:

5 Storage Class                      - Specify Data Set Performance and Availability



As with the Data Class, you can use the Storage Class panel to list, display, define, and alter entries. See Figure 5-21.

```

STORAGE CLASS APPLICATION SELECTION
Command ==>

To perform Storage Class Operations, Specify:
  CDS Name . . . . . 'SYS1.SMS.SCDS'
                                     (1 to 44 character data set name or 'Active' )
  Storage Class Name . . DB9ASDR   (For Storage Class List, fully or
                                     partially specified or * for all)

Select one of the following options :
  4 1. List           - Generate a list of Storage Classes
    2. Display        - Display a Storage Class
    3. Define         - Define a Storage Class
    4. Alter          - Alter a Storage Class
    5. Cache Display  - Display Storage Classes/Cache Sets
    6. Lock Display   - Display Storage Classes/Lock Sets

If List Option is chosen,
  Enter "/" to select option          Respecify View Criteria
                                     Respecify Sort Criteria

If Cache Display is Chosen, Specify Cache Structure Name . .
If Lock Display is Chosen, Specify Lock Structure Name . . .

```

Figure 5-21 Initial Storage Class panel

Striping is enabled on page 1 (Figure 5-22 on page 260) with the Sustained Data Rate (SDR). More information about striping is available in the striping section. For guaranteed space, if the SDR value is 1 or greater, it is ignored; in which case, the target number of stripes is the greater of either the volume count that is specified or the number of specified volume serial numbers. If Guaranteed Space is not enabled, the number of stripes is the SDR/4, which in this case 12/4=3 stripes. The number of stripes must correspond to the number of volumes, three stripes requires three volumes. Tests have shown that increased performance stripes should be in even numbers, the odd numbers displayed are examples only.

For more information, see *DFSMS/MVS DFSMSdfp Storage Administration Reference*, SC26-7402 (the “Rules for Striping Volume selection” section in the “Defining Storage Classes” chapter).

```

STORAGE CLASS ALTER                               Page 1 of 2
Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS
Storage Class Name  : DB9ASDR
To ALTER Storage Class, Specify:
  Description ==>
    ==>
  Performance Objectives
    Direct Millisecond Response . . . . . (1 to 999 or blank)
    Direct Bias . . . . . (R, W or blank)
    Sequential Millisecond Response . . . . . (1 to 999 or blank)
    Sequential Bias . . . . . (R, W or blank)
    Initial Access Response Seconds . . . . . (0 to 9999 or blank)
    Sustained Data Rate (MB/sec) . . . . . 12 (0 to 999 or blank)
    OAM Sublevel . . . . . (1, 2 or blank)
    Availability . . . . . N (C, P ,S or N)
    Accessibility . . . . . N (C, P ,S or N)
    Backup . . . . . (Y, N or Blank)
    Versioning . . . . . (Y, N or Blank)

```

Figure 5-22 Storage Class: Page 1

The number of volumes that are required is usually not known by most people working with DB2. If you have a 900-cylinder active log data set on one volume and now want to stripe it across three volumes, three volumes are required each with 300 cylinders.

Page 1 shows such entries as Availability and Accessibility, and they are both set to N for No preference. However, modern devices already include features such as dual copy and concurrent copy, and therefore do not require the additional SMS setting.

Page 2 (Figure 5-23) has entries for enabling Guaranteed Space and Multi-Tiered SG (storage groups). Multitiered storage groups are described further in 5.2.2, “Storage Class implementation” on page 258. If you use them, however, set them in the Storage Class.

```

STORAGE CLASS ALTER                               Page 2 of 2
Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS
Storage Class Name  : DB9ASDR
To ALTER Storage Class, Specify:

  Guaranteed Space . . . . . N (Y or N)
  Guaranteed Synchronous Write . . . N (Y or N)
  Multi-Tiered SG . . . . . (Y, N, or blank)
  Parallel Access Volume Capability N (R, P, S, or N)
  CF Cache Set Name . . . . . (up to 8 chars or blank)
  CF Direct Weight . . . . . (1 to 11 or blank)
  CF Sequential Weight . . . . . (1 to 11 or blank)
  CF Lock Set Name . . . . . (up to 8 chars or blank)

```

Figure 5-23 Storage Class: Page 2 of 2

Guaranteed Space means just that: if you ask for 100 cylinders you must get 100 cylinders. This reason is why Guaranteed Space cannot be used with the Data Class specification of Space Constraint Relief which can reduce an allocation and redrive the request. Guaranteed Space also means guaranteed VOLSER. If you specify one or more specific volumes, those are the only ones eligible for the request. If you have a storage group with 100 volumes with VOLSERS from DB2001 - DB2100 and you enable the use of Guaranteed Space and ask for volume DB2043, the only eligible volume for this request is DB2043. If Guaranteed Space is not enabled, all 100 volumes are eligible. For multi-volume data sets, Guaranteed Space allocates the primary allocation on all volumes, not only the first volume, such as when Guaranteed Space is not enabled.

- ▶ When using SMS to allocate DB2 objects, the DB2 STOGROUP allows you to specify a certain volume. For allocations that are not using Guaranteed Space, specify “\*” (one asterisk) only for VOLUMES. Although if you want, you may use a specific VOLSER when not using Guaranteed Space, there are times when this approach can cause unintended problems and should be avoided. Code the “\*” (with only one asterisk in the DB2 STOGROUP) when not using Guaranteed Space.

At times, a storage administrator sets up the SMS ACS routines to check for specific volume types for allocation. Much of the time an \* (asterisk) is not provided for the test. Without the ACS routine having an \* (asterisk) as part of a valid test, your allocation can fail. If you will be creating STOGROUPs with VOLUMES (“\*”), make sure your storage administrator adds an \* (asterisk) to the test of device allocations in the ACS routines.

- ▶ When DB2 data sets are SMS-managed, do not use a non-DB2 volume for the DB2 STOGROUP because it will not give DB2 control of the request and will not prevent SMS from allocating the data set to a SMS-managed volume. Even though a non-SMS volume was chosen, the allocation will continue and choose an appropriate SMS-managed volume.

Generally, for the DB2 environment, the Separation Profile or Guaranteed Space is used for specific data sets only, specifically to manually place the active log and boot strap data sets. All other DB2 data sets should not use the Guaranteed Space attribute. Hot spots have been greatly reduced by such breakthroughs as PAV, and storage pool striping.

Before using Guaranteed Space, consider the drawbacks, such as limiting your request to specific volumes even when there may be an enormous amount of space available elsewhere in the storage group, not being able to use the Data Class Space Constraint Removal reduction, and having the primary allocation on all volumes for a multi-volume data set.

At times, there are performance benefits to striping the active log and archive log data sets. The active log data sets are LDSs and uses VSAM striping. In DB2 9 NFM, you can stripe and compact (compress) the archive log data sets, which were converted from BDAM to BSAM, therefore eligible for striping. Although VSAM and sequential striping have differences, the way to implement them through SMS is the same.

Example 5-12 adds a new DB2 *active log data set* that is 4 GB (actually slightly less at 4095 MB).

*Example 5-12 Striping active log data sets*

---

```

DEFINE CLUSTER -
  ( NAME (DB9AU.LOGCOPY1.DS04) -
    DATACLASS(DB2EFEAS) -
    STORAGECLASS(DB9ASDR) -
    VOLUMES(SBOX1Y,SBOX1X,SBOX0H) -
      MB(4095,0) -
      LINEAR )
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IGD17070I DATA SET DB9AU.LOGCOPY1.DS04
ALLOCATED SUCCESSFULLY WITH 3 STRIPE(S).
IGD17172I DATA SET DB9AU.LOGCOPY1.DS04
IS ELIGIBLE FOR EXTENDED ADDRESSABILITY
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX1V IS 0
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX1W IS 0
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX0T IS 0
IDC0512I NAME GENERATED-(D) DB9AU.LOGCOPY1.DS04.DATA
IDC0181I STORAGECLASS USED IS DB9ASDR
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0181I DATACLASS USED IS DB2EFEAS
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

DEFINE CLUSTER -
  ( NAME (DB9AU.LOGCOPY2.DS04) -
    DATACLASS(DB2EFEAS) -
    STORAGECLASS(DB9ASDR) -
    VOLUMES(SBOX1Y,SBOX1X,SBOX0H) -
      MB(4095,0) -
      LINEAR )
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IGD17070I DATA SET DB9AU.LOGCOPY2.DS04
ALLOCATED SUCCESSFULLY WITH 3 STRIPE(S).
IGD17172I DATA SET DB9AU.LOGCOPY2.DS04
IS ELIGIBLE FOR EXTENDED ADDRESSABILITY
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX1W IS 0
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX0F IS 0
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX0T IS 0
IDCAMS SYSTEM SERVICES
IDC0512I NAME GENERATED-(D) DB9AU.LOGCOPY2.DS04.DATA
IDC0181I STORAGECLASS USED IS DB9ASDR
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0181I DATACLASS USED IS DB2EFEAS
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0

```

---

Note the following information:

- ▶ Data Class DB2EFEAS was used for the EF attribute.
- ▶ Data Class DB2EFEAS also enabled EA. EA is not required because the data set did not exceed 4 GB. The Data Class already had EF and EA enabled, and there was no reason not to use this Data Class.

- Storage Class DB9ASDR was used. Regarding DB9ASDR, the examples show Storage Class has no Guaranteed Space, but has an SDR value of 12. Therefore because Guaranteed Space is not enabled, the result is three stripes and therefore 3 volumes.
- Volumes SBOX1Y, SBOX1X, SBOX0H are specified for both data set allocations. Storage Class DB9ASDR does not have Guaranteed Space enabled; during allocation SMS chose to place the data set for LOGCOPY1 on volumes SBOX1V, SBOX1W, SBOX0T, and for LOGCOPY2 on volumes SBOX1W, SBOX0F, SBOX0T instead.

**Note:** Because Guaranteed Space is not enabled for these data sets, they both share two of the three volumes: SBOX1W and SBOX0T. Because they are the two software duplexed versions of the same *active log data set*, they should have been allocated on separate physical disk drives, if not different disk controllers. This allocation should be resolved before continuing by deleting and defining the data sets using the Separation Profile or Guaranteed Space

- A value of 4095 MB is used instead of 4096 MB which is the 4 GB limit. When 4096 MB is specified, formatting the new data sets results in zero-formatted rows. APAR PK93879 resolves this problem and 4096 MB can be requested (be sure to use DSNJLOGF to format the new active log data set.).

Example 5-13 shows the LISTCAT output for the logcopy1 data set that is newly defined on SBOX1Y. The logcopy2 data set gets defined on volume SBOX1W.

*Example 5-13 LISTCAT output for DB9AU.LOGCOPY1.DS04*

```

OCLUSTER ----- DB9AU.LOGCOPY1.DS04
  IN-CAT --- UCAT.DB9A
  HISTORY
    DATASET-OWNER----- (NULL)      CREATION-----2010.140
    RELEASE-----2          EXPIRATION-----0000.000
  SMSDATA
    STORAGECLASS ----DB9ASDR      MANAGEMENTCLASS---MCDB22
    DATACLASS  -----DB2EFEAS    LBACKUP ---0000.000.0000

  ASSOCIATIONS
    DATA-----DB9AU.LOGCOPY1.DS04.DATA
0  DATA ----- DB9AU.LOGCOPY1.DS04.DATA

    CLUSTER--DB9AU.LOGCOPY1.DS04
  ATTRIBUTES
    KEYLEN-----0          AVGLRECL-----0          BUFSPACE-----8192
  CFSIZE-----4096
    RKP-----0          MAXLRECL-----0          EXCPEXIT----- (NULL)
  CI/CA-----180
    STRIPE-COUNT-----3
  SHROPTNS(1,3)  RECOVERY  UNIQUE          NOERASE          LINEAR          NOWRITECHK
  NOIMBED        NOREPLICAT
    UNORDERED        NOREUSE        NONSPANNED        EXTENDED        EXT-ADDR
  STATISTICS
    REC-TOTAL-----0          SPLITS-CI-----0          EXCPS-----23295
    REC-DELETED-----0        SPLITS-CA-----0          EXTENTS-----4
    REC-INSERTED-----0        FREESPACE-%CI-----0        SYSTEM-TIMESTAMP:
    REC-UPDATED-----0        FREESPACE-%CA-----0          X'C601D0C46F959D0B'
    REC-RETRIEVED-----0      FREESPC-----0
  ALLOCATION
    SPACE-TYPE-----TRACK      HI-A-RBA-----4293918720
    SPACE-PRI-----87360      HI-U-RBA-----4293918720
    SPACE-SEC-----0

```

VOLUME		
0	<b>VOLSER-----SBOX1V</b>	PHYREC-SIZE-----4096      HI-A-RBA-----4293918720
	<b>EXTENT-NUMBER-----1</b>	
	DEVTYPE-----X'3010200F'	PHYRECS/TRK-----12      HI-U-RBA-----4293918720
	EXTENT-TYPE-----X'00'	
	VOLFLAG-----PRIME	<b>TRACKS/CA-----5</b>
	<b>STRIPE-NUMBER-----1</b>	
	EXTENTS:	
	LOW-CCHH-----X'46550000'	LOW-RBA-----0 <b>TRACKS-----29120</b>
	HIGH-CCHH-----X'4DEA0004'	HIGH-RBA-----4293918719
VOLUME		
	<b>VOLSER-----SBOX1W</b>	PHYREC-SIZE-----4096      HI-A-RBA-----4293918720
	<b>EXTENT-NUMBER-----1</b>	
	DEVTYPE-----X'3010200F'	PHYRECS/TRK-----12      HI-U-RBA-----0
	EXTENT-TYPE-----X'00'	
	VOLFLAG-----PRIME	<b>TRACKS/CA-----5</b>
	<b>STRIPE-NUMBER-----2</b>	
	EXTENTS:	
	LOW-CCHH-----X'288B0000'	LOW-RBA-----0 <b>TRACKS-----29120</b>
	HIGH-CCHH-----X'30200004'	HIGH-RBA-----4293918719
VOLUME		
	<b>VOLSER-----SBOX0T</b>	PHYREC-SIZE-----4096      HI-A-RBA-----4293918720
	<b>EXTENT-NUMBER-----2</b>	
	DEVTYPE-----X'3010200F'	PHYRECS/TRK-----12      HI-U-RBA-----0
	EXTENT-TYPE-----X'00'	
	VOLFLAG-----PRIME	<b>TRACKS/CA-----5</b>
	<b>STRIPE-NUMBER-----3</b>	
	EXTENTS:	
	LOW-CCHH-----X'15990000'	LOW-RBA-----0 <b>TRACKS-----7685</b>
	HIGH-CCHH-----X'17990004'	HIGH-RBA-----1133199359
	LOW-CCHH-----X'1A800000'	LOW-RBA-----1133199360 <b>TRACKS-----21435</b>
	HIGH-CCHH-----X'2014000E'	HIGH-RBA-----4293918719

Note the following information about the LISTC output:

- ▶ We verify that the proper Data Class and Storage Class are used.
- ▶ Extended is one of the attributes that is listed.
- ▶ Three stripes are allocated, one for each volume. There are four extents because the allocation on volume SBOX0T requires the two extents, because there is not enough single extent free space for 29,190 tracks.
- ▶ The allocation is in tracks, also noted by TRACKS/CA. The 4095 MB request translates to 87,360 tracks (5,824 cylinders) total. Striping requires that each stripe be the same size, therefore each stripe contains 29,120 tracks on each volume. Even though the stripe on volume SBOX0T allocated in two extents, the results are the same number of required tracks 7685+21435=29120.
- ▶ The default attributes RECOVERY and NOREUSE have been used:
  - DB2 never attempts to open an active log with RESET, therefore REUSE/NOREUSE makes no difference.
  - SPEED or RECOVERY are not honored by media manager. Regardless of the setting, DB2 preformats CIs if the active log is not already preformatted. The DSNJLOGF might run faster if SPEED is set. If a new active log is not formatted by DSNJLOGF before DB2 uses it, the formatting takes place as the log is written, creating overhead only the first time that active log data set is used.

For the *archive log data sets* that correspond to this new active log, there are two options:

- ▶ Use large format sequential data sets.
- ▶ Use the EF attribute.

These two options are mutually exclusive. Because striping is required in our example, large format sequential is not used. As with the active log data sets, the Data Class requires EF to be enabled, and the Storage Class requires an SDR value. In this case, because compaction is requested, the Data Class must enable compaction also. Both values are set on page 2 of 5 of the Data Class. See Figure 5-24.

DATA CLASS ALTER Command ==>	Page 2 of 5
SCDS Name . . . : SYS1.SMS.SCDS Data Class Name : DB9AARCH  To ALTER Data Class, Specify:	
<b>Data Set Name Type . . . . . EXT</b>	(EXT, HFS, LIB, PDS, Large or blank)
If Ext . . . . . R	(P=Preferred, R=Required or blank)
Extended Addressability . . N	(Y or N)
Record Access Bias . . . . U	(S=System, U=User or blank)
Space Constraint Relief . . . N	(Y or N)
Reduce Space Up To (%) . .	(0 to 99 or blank)
Dynamic Volume Count . . .	(1 to 59 or blank)
<b>Compaction . . . . . Y</b>	(Y, N, T, G or blank)
Spanned / Nonspanned . . . .	(S=Spanned, N=Nonspanned or blank)
System Managed Buffering . .	(1K to 2048M or blank)
System Determined Blocksize N	(Y or N)
EATTR . . . . .	(0=Opt, N=No or blank)

Figure 5-24 Alter of Data Class DB9AARCH: Enable EF and compaction

Unlike the active log data sets, the SMS classes for the DB2 *archive log data sets* cannot be requested by the user. Allocation is through DSNZPARMs shown in Example 5-14.

Example 5-14 DSNZPARMs used for our test of the archive log data sets: allocation

DSN6ARVP	ALCUNIT=CYL, ARCPFX1=DB9AU.ARCHLOG1, ARCPFX2=DB9AU.ARCHLOG2, PRIQTY=5826, SECQTY=180, SVOLARC=NO, UNIT=SYSALLDA, UNIT2=	X X X X X X X
----------	--	---------------------------------

Allocations of the archive log data sets are based on the amount of space that is used in the active log data set. If you started using a 4 GB active log and issued DB2 command ARCHIVE LOG, only a small archive log data set would be created, not based on the specific DSNZPARM specifications for PRIQTY and SECQTY.

DSNZPARM parameter COMPACT is not the same as SMS COMPACTION. The DSNZPARM COMPACT controls whether Improved Data Recording Capability (IDRC) is used for 3480 or 3490 tape drives.

### Example 5-15 Data and Storage Class examples for the archive log data sets

Example 5-16 shows the LISTCAT output for the archive log data set.

*Example 5-16 LISTCAT and ISPF 3.4 of archive log data set striped and compacted*

```
General Data                                Current Allocation
Management class . . : MCDB22              Allocated cylinders : 3,681
```





*Example 5-17 LISCAT and ISPF 3.4 of archive log data set striped, with compaction disabled*

Data Set Name . . . . : DB9AU.ARCHLOG1.A0000949

```
Esssssssssss Volume Information sssssssssssN
e
e Command ==>
e
e All allocated volumes:
e                                     More:      +
e   Number of volumes allocated: 3
e
e SB0X1X  SB0X1W  SB0X1V
e
e
e
e
DssssssssssssssssssssssssssssssssssssssssM
```

268 DB2 9 for z/OS and Storage Management

Notice the significant difference in size, because of compression, between Example 5-17 on page 268 and Example 5-16 on page 266. When uncompressed, the allocation size is about the same as the active log data set: 5,824 cylinders. Compare this value when compressed: 3,680 cylinders. We can also determine compaction from the values in bold in Example 5-16 on page 266:

```
USER-DATA-SIZE-----4293918720
COMP-USER-DATA-SIZE-----3085131814
```

Compressing the data set saved 2,144 cylinders.

### 5.2.3 Management Class implementation

Management Class can be used for a variety of functions:

- ▶ Expire data sets
- ▶ Indicate whether a user or Data Class can specify retention period
- ▶ Partial release of data sets. For DB2 LDSs, partial release is for data sets with EF enabled, but without Guaranteed Space.
- ▶ Migrate to Level 1 for data sets not used for a specific time
- ▶ Migrate to Level 2 for data sets not used for a specific time
- ▶ Migrate from Level 1 to Level 2 for data sets not used for a specific time
- ▶ Specify whether migration should be automatic, by command, or both
- ▶ Migrate based on the number of GDG objects
- ▶ Determine action for rolled off GDG objects
- ▶ Specify different interval backup attributes
- ▶ Show criteria for ABARS

Enter the Management Class by selecting ISMF option **3**:

```
3  Management Class           - Specify Data Set Backup and Migration Criteria
```

The initial panel, shown in Figure 5-26 on page 270, page 1, is similar to the other SMS Class panels.

```

MANAGEMENT CLASS APPLICATION SELECTION          Page 1 of 2
Command ==>

To perform Management Class Operations, Specify:
  CDS Name . . . . . 'SYS1.SMS.SCDS'
                                     (1 to 44 character data set name or 'Active' )
  Management Class Name . . . *      (For Management Class List, fully or
                                     partially specified or * for all)

Select one of the following options :
  1. List    - Generate a list of Management Classes
  2. Display - Display a Management Class
  3. Define  - Define a Management Class
  4. Alter   - Alter a Management Class

If List Option is chosen,
  Enter "/" to select option      Respecify View Criteria
                                   Respecify Sort Criteria

```

Figure 5-26 Initial Management Class page 1

Some customers migrate their DB2 LDSs, most customer do not, especially not in their production environments. When migrating DB2 LDSs, verify the ZPARM setting for RECALL and RECALLD to avoid DB2 thread failures. For production environments, DFSMSShsm is primarily used to migrate image copy and archive log data sets allocated to disk to ML1, ML2, or both. Customers find the release of unused space in their non-production LDSs useful, saving a significant amount of space.

Figure 5-27 shows a Management Class called DB9AML1, and a storage group called DB9AML1 that has one mod 3 volume, JIAB1C. DFSMSShsm interval migration was enabled for the LPAR and storage group DB9AML1 (see Figure 5-28 on page 271).

```

MANAGEMENT CLASS ALTER          Page 2 of 5
Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS
Management Class Name : DB9AML1

To ALTER Management Class, Specify:

  Partial Release . . . . . N          (Y, C, YI, CI or N)

Migration Attributes
  Primary Days Non-usage . . . . 0      (0 to 9999 or blank)
  Level 1 Days Non-usage . . . . 60    (0 to 9999, NOLIMIT or blank)
  Command or Auto Migrate . . . . BOTH (BOTH, COMMAND or NONE)

GDG Management Attributes
  # GDG Elements on Primary . . .      (0 to 255 or blank)
  Rolled-off GDS Action . . . . .      (MIGRATE, EXPIRE or blank)

```

Figure 5-27 Alter the Management Class DB9AML1

Only page 2 of 5 is illustrated because it is the only one modified. Page 2 shows that data sets on primary volumes are migrated after 0 (zero) days to ML1, then from ML1 to ML2 after 60 days. Migration can be done by command or auto-migration.

DB9AML1 was set with a HIGH of 20% and a low of 0% to encourage quick migration of the image copy data sets. See Figure 5-28.

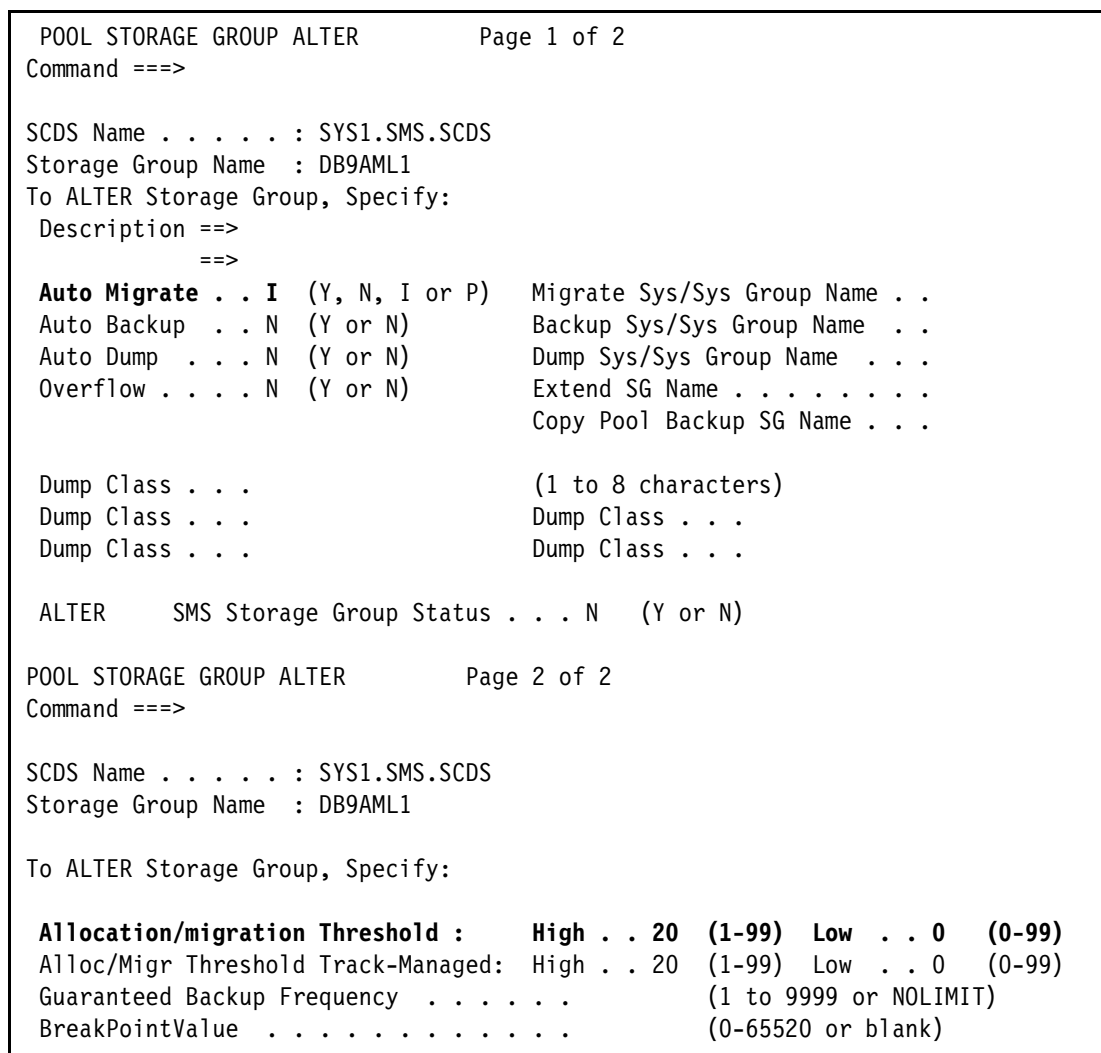


Figure 5-28 Alter the storage group DB9AML1

The data set name pattern for allocation to DB9AML1 is as follows (Example 5-18):

DB9AU.IC.JOHN.\*\*

Example 5-18 ACS routines for migration of image copy data sets

```

PROC STORCLAS
FILTLIST DB9AML1 INCLUDE(DB9AU.IC.JOHN.**)
  IF &DSN = &DB9AML1
  THEN DO
    SET &STORCLAS = 'DB9AML1'
  EXIT
END
END
  
```



## VOL SER NOS= JIAB1C

IGD17380I STORAGE GROUP (DB9AML1) IS ESTIMATED AT 88% OF CAPACITY,  
WHICH EXCEEDS ITS HIGH ALLOCATION THRESHOLD OF 20%

ARC0580I INTERVAL MIGRATION STARTING AT 13:00:00 ON 628  
ARC0580I (CONT.) 2010/05/28  
ARC0522I SPACE MANAGEMENT STARTING ON VOLUME 629  
ARC0522I (CONT.) JIAB1C(SMSI) AT 13:00:00 ON 2010/05/28, SYSTEM SC63  
ARC0523I SPACE MANAGEMENT ENDED ON VOLUME JIAB1C, 630  
ARC0523I (CONT.) 00000003 DATA SET(S) MIGRATED/DELETED, 0000043726  
ARC0523I (CONT.) TRACK(S) FREED, MINAGE 0, TIME 13:01:06  
ARC0581I INTERVAL MIGRATION ENDED SUCCESSFULLY AT 633  
ARC0581I (CONT.) 13:01:06 ON 2010/05/28

This volume, JIAB1C, is used for *image copy data sets*. The image copy of the A001 data set consumes almost all of the mod 3. After the data set is allocated, SMS issues message IGD17380I, which states that the HIGH value of 20% was exceeded. After allocation, DFSMSHsm interval migration starts and migrates the image copy data set and the two “dummy” data sets.

Figure 5-29 lists the migrated volume.

DSLIS - Data Sets Matching DB9AU.IC.JOHN			Row 1 of 1
Command ==>			Scroll ==> PAGE
Command - Enter "/" to select action	Message	Volume	
-----			-----
DB9AU.IC.JOHN.TEST8			MIGRAT1

Figure 5-29 ISPF 3.4 of the migrated image copy data set

The data set can be recalled and brought back to disk.

## 5.2.4 Storage group implementation

Note the following information about storage groups:

- ▶ Contain VOLSERS; a volume can belong to only one storage group.
- ▶ Determine whether DFSMSHsm migration, dump, or incremental backup should be enabled.
- ▶ Allow a storage group to be an overflow volume.
- ▶ Allow a storage group to be an extended storage group. An overflow storage group can also be an extended storage group.
- ▶ Specify a Copy Pool backup storage group name.
- ▶ Contain HIGH and LOW values for the storage group.
- ▶ Specify a Break Point Value (BPV) for EAV devices.

Storage Group is not an option from ISMF for users, you must be signed in as a storage administrator. While in Storage Administrator mode, select option 6 for Storage Group:

6 Storage Group - Specify Volume Names and Free Space Thresholds

The initial storage group panel opens (Figure 5-30).

```
STORAGE GROUP APPLICATION SELECTION
Command ==>

To perform Storage Group Operations, Specify:
CDS Name . . . . . 'SYS1.SMS.SCDS'
                                     (1 to 44 character data set name or 'Active' )
Storage Group Name    DB9A*          (For Storage Group List, fully or
                                     partially specified or * for all)
Storage Group Type     (VIO, POOL, DUMMY, COPY POOL BACKUP,
                                     OBJECT, OBJECT BACKUP, or TAPE)

Select one of the following options :
1 1. List   - Generate a list of Storage Groups
  2. Define - Define a Storage Group
  3. Alter  - Alter a Storage Group
  4. Volume - Display, Define, Alter or Delete Volume Information

If List Option is chosen,
Enter "/" to select option      Respecify View Criteria
                                Respecify Sort Criteria
```

*Figure 5-30 Storage group: initial panel*

The initial panel for the storage group differs from the class panels; there is no DISPLAY option. The closest to display is the Alter option.

As noted in the Storage Group Type, the classifications are VIO, POOL, DUMMY, COPY POOL BACKUP, OBJECT, OBJECT BACKUP, or TAPE. This section focuses only on POOL. DB2 objects are created in the POOL classification.



The Define and Alter panels are the same and have only two primary pages shown respectively in Figure 5-31 and Figure 5-32.

POOL STORAGE GROUP ALTER		Page 1 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
Storage Group Name : DB9A		
To ALTER Storage Group, Specify:		
Description ==> DB9A DATA SETS		
==>		
Auto Migrate . . . N	(Y, N, I or P)	Migrate Sys/Sys Group Name . .
Auto Backup . . . N	(Y or N)	Backup Sys/Sys Group Name . .
Auto Dump . . . N	(Y or N)	Dump Sys/Sys Group Name . . .
Overflow . . . . N	(Y or N)	Extend SG Name . . . . .
		Copy Pool Backup SG Name . . .
Dump Class . . .		(1 to 8 characters)
Dump Class . . .		Dump Class . . .
Dump Class . . .		Dump Class . . .
ALTER SMS Storage Group Status . . . N (Y or N)		

Figure 5-31 Storage Group Alter: page 1 of 2

POOL STORAGE GROUP ALTER		Page 2 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
Storage Group Name : DB9A		
To ALTER Storage Group, Specify:		
Allocation/migration Threshold :	High . . 99	(1-99) Low . . (0-99)
Alloc/Migr Threshold Track-Managed:	High . . 85	(1-99) Low . . 1 (0-99)
Guaranteed Backup Frequency . . . . .		(1 to 9999 or NOLIMIT)
BreakPointValue . . . . .		(0-65520 or blank)

Figure 5-32 Storage Group Alter: page 2 of 2

High and Low are important parameters. Thresholds refer to the percentage used for a volume. Remember that, although you can mix the same unit types (but various emulated disk sizes) the percentage can make a big difference. If you have one Model 3 and one Model 9 with 54 GB (mod 54), then 80% of each is an enormous difference between the two volumes. If you do not use DFSMSShm to migrate data, LOW is not used and does not need to be set. HIGH is important and the value depends on the use of the storage group.

Specifying Y for SMS storage group Status on page 1 allows you to alter to which LPARs the storage group is online (Figure 5-33).

SMS STORAGE GROUP STATUS ALTER

Page 1 of 2

Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS

Storage Group Name : DB9A

Storage Group Type : POOL

To ALTER Storage Group System/

Sys Group Status, Specify:

System/Sys

SMS SG

System/Sys

SMS SG

Group Name

Status

Group Name

Status

-----

-----

-----

-----

SC63

==>

ENABLE

SC64

==>

ENABLE

SC65

==>

ENABLE

SC70

==>

ENABLE

==>

==>

==>

==>

==>

==>

( Possible SMS SG Status

for each:

- Pool SG Type

NOTCON, ENABLE, DISALL

DISNEW, QUIALL, QUINEW

- Tape SG Type

NOTCON, ENABLE,

DISALL, DISNEW

- Copy Pool Backup SG Type

NOTCON, ENABLE )

\* SYS GROUP = sysplex

minus Systems in the

Sysplex explicitly

defined in the SCDS

Figure 5-33 Panel to specify SMS Storage Group Status

Option 4 of the Storage Group panel is used to work with volumes:

4. Volume - Display, Define, Alter or Delete Volume Information

For example, you might want to add 10 volumes, JOHN00 - 09, as shown in Figure 5-34.

STORAGE GROUP VOLUME SELECTION

Command ==>

CDS Name . . . . . : SYS1.SMS.SCDS

Storage Group Name : DB9A

Storage Group Type : POOL

Select One of the following Options:

2 1. Display - Display SMS Volume Statuses (Pool & Copy Pool Backup only)

2. Define - Add Volumes to Volume Serial Number List

3. Alter - Alter Volume Statuses (Pool & Copy Pool Backup only)

4. Delete - Delete Volumes from Volume Serial Number List

Specify a Single Volume (in Prefix), or Range of Volumes:

Prefix From To Suffix Type

==>

JOHN

00

09

-

==>

==>

==>

Figure 5-34 Page 1 of 2 to add volumes

Page 1 (in Figure 5-34 on page 276) does not have a page number, page 2 actually displays “Page 1 of 2” (Figure 5-35). Page 2 is only a continuation of page 1 and allows you to add volumes to different LPARs.

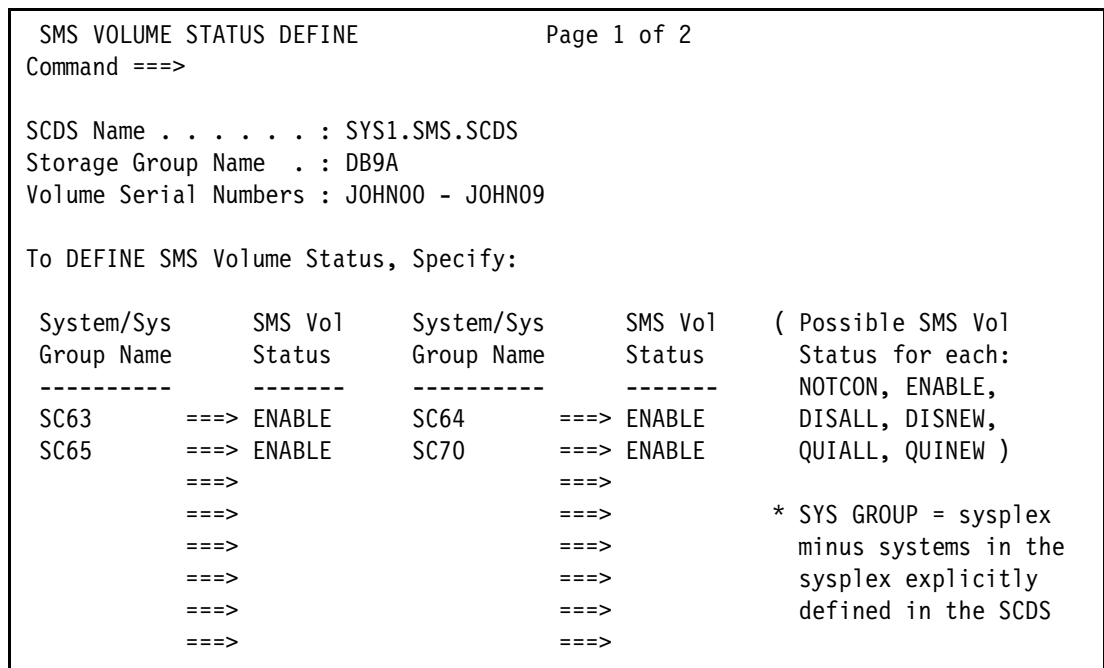


Figure 5-35 Page 2 of 2 to add volumes

Notice the various SMS volume status definitions, by volume, for each LPAR:

NOTCON, ENABLE, DISALL, DISNEW, QUIALL, QUINEW

The most common settings are ENABLE and QUINEW:

- ▶ ENABLE allows the volume to be used.
- ▶ QUINEW prevents the system from scheduling new allocations to a volume unless no other volumes meet the space requested.

For a full description of the status settings and how to use the prefix, suffix, and so on for the volumes, see *DFSMSdfp Storage Administration*, SC26-7402.

Extend Storage Groups allow data sets that have already been created to be extended to a separate storage group if space is not available in the original storage group. For example, you have one storage group for table spaces, and one for indexes. This technique is no longer required with the proper use of PAV, but can still be used. You can use Extend Storage Groups to each other so that the index storage group would be available for the table spaces and the table spaces storage groups would be available for indexes. Overflow storage groups can be used as Extend Storage Groups. Using Extend Storage Groups is for availability. The data set may not extend within the same storage group already allocated, however the extend will not fail and cause an outage.

For example, we have two Storage Groups: DB9ATST1 and DB9ATST2. Both contain only one volume and are defined as an Extend Storage Group to each other:

- ▶ DB9ATST1 contains volume JIAA10 with data set DB9AU.DSNDB%.\*.TEST1\*.\*\* pattern
- ▶ DB9ATST2 contains volume JIAB10 with data set DB9AU.DSNDB%.\*.TEST2\*\*.\*\* pattern.

Using an alter process of storage group DB9ATST1 is shown in Figure 5-36.

POOL STORAGE GROUP ALTER		Page 1 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
<b>Storage Group Name : DB9ATST1</b>		
To ALTER Storage Group, Specify:		
Description ==>		
==>		
Auto Migrate . . N	(Y, N, I or P)	Migrate Sys/Sys Group Name . .
Auto Backup . . N	(Y or N)	Backup Sys/Sys Group Name . .
Auto Dump . . . N	(Y or N)	Dump Sys/Sys Group Name . . .
Overflow . . . . N	(Y or N)	<b>Extend SG Name . . . . . DB9ATST2</b>
		Copy Pool Backup SG Name . . .
Dump Class . . .		(1 to 8 characters)
Dump Class . . .		Dump Class . . .
Dump Class . . .		Dump Class . . .
ALTER SMS Storage Group Status . . . N (Y or N)		
POOL STORAGE GROUP ALTER		Page 2 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
Storage Group Name : DB9ATST1		
To ALTER Storage Group, Specify:		
Allocation/migration Threshold : High . . 85 (1-99) Low . . 1 (0-99)		
Alloc/Migr Threshold Track-Managed: High . . 85 (1-99) Low . . 1 (0-99)		
Guaranteed Backup Frequency . . . . . (1 to 9999 or NOLIMIT)		
BreakPointValue . . . . . (0-65520 or blank)		

Figure 5-36 Storage Group DB9ATST1: pages 1 and 2

Alter using an alter process of storage group DB9ATST2 is shown in Figure 5-37.

POOL STORAGE GROUP ALTER

Page 1 of 2

Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS

**Storage Group Name : DB9ATST2**

To ALTER Storage Group, Specify:

Description ==>

==>

Auto Migrate . . N (Y, N, I or P)

Auto Backup . . N (Y or N)

Auto Dump . . . N (Y or N)

Overflow . . . . N (Y or N)

Migrate Sys/Sys Group Name . .

Backup Sys/Sys Group Name . .

Dump Sys/Sys Group Name . . .

**Extend SG Name . . . . . DB9ATST1**

Copy Pool Backup SG Name . . .

Dump Class . . .

Dump Class . . .

Dump Class . . .

(1 to 8 characters)

Dump Class . . .

Dump Class . . .

ALTER SMS Storage Group Status . . . N (Y or N)

POOL STORAGE GROUP ALTER

Page 2 of 2

Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS

Storage Group Name : DB9ATST2

To ALTER Storage Group, Specify:

Allocation/migration Threshold : High . . 85 (1-99) Low . . 1 (0-99)

Alloc/Migr Threshold Track-Managed: High . . 85 (1-99) Low . . 1 (0-99)

Guaranteed Backup Frequency . . . . . (1 to 9999 or NOLIMIT)

BreakPointValue . . . . . (0-65520 or blank)

Figure 5-37 Storage Group DB9ATST2: pages 1 and 2

Line operator command LISTVOL lists the volumes associated with each storage group.  
LISTVOL output for storage group DB9ATST1 is shown in Figure 5-38.

VOLUME LIST

Command ==>

Scroll ==> PAGE

Entries 1-1 of 1

Data Columns 3-8 of 43

Enter Line Operators below:

LINE	VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
OPERATOR	SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
---(1)----	-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
	JIAA10	221K	0	2771279K	0	221K	1

Figure 5-38 LISTVOL output for storage group DB9ATST1

LISTVOL output for storage group DB9ATST2 is shown in Figure 5-39.

VOLUME LIST							
Command ==>				Scroll ==> PAGE			
Enter Line Operators below:				Entries 1-1 of 1			
				Data Columns 3-8 of 43			
LINE	VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
OPERATOR	SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
---(1)---	-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
	JIAB10	2737690K	99	33810K	0	2737469K	2

Figure 5-39 LISTVOL output for storage group DB9ATST2

Both volumes are mod 3s and were empty before allocating any data sets. Only the extend from DB9ATST1 to DB9ATST2 is illustrated, not the other way around.

Example 5-20 shows the ACS routine used for the example.

Example 5-20 Storage Class and Storage Group ACS routines for TEST1 and TEST2 data sets

---

#### PROC STORCLAS

```
FILTLIST DB9ATST1 INCLUDE(DB9AU.DSNDB%.*.TEST1*.*?)
FILTLIST DB9ATST2 INCLUDE(DB9AU.DSNDB%.*.TEST2*.*?)
  IF &DSN = &DB9ATST1
  THEN DO
    SET &STORCLAS = 'DB9ATST1'
  EXIT
  END
  IF &DSN = &DB9ATST2
  THEN DO
    SET &STORCLAS = 'DB9ATST2'
  EXIT
  END
```

#### PROC STORGRP

```
SELECT
  WHEN (&STORCLAS = 'DB9ATST1')
  SET &STORGRP = 'DB9ATST1'
  WHEN (&STORCLAS = 'DB9ATST2')
  SET &STORGRP = 'DB9ATST2'
END
END
```

---

Three data sets are created in DB9ATST1. See Example 5-21.

*Example 5-21 TEST1 data sets*

Command - Enter "/" to select action	Message	Volume
DB9AU.DSNDBD.DSNDB04.TEST1HOL.I0001.A001		JIAA10
DB9AU.DSNDBD.DSNDB04.TEST1ONE.I0001.A001		JIAA10
DB9AU.DSNDBD.DSNDB04.TEST1TWO.I0001.A001		JIAA10+
DATA ----- DB9AU.DSNDBD.DSNDB04.TEST1TWO.I0001.A001		
ALLOCATION		
SPACE-TYPE-----CYLINDER	HI-A-RBA-----	2211840
SPACE-PRI-----1	HI-U-RBA-----	2211840
SPACE-SEC-----1		
VOLUME		
VOLSER-----JIAA10	PHYREC-SIZE-----	4096 HI-A-RBA-----737280
EXTENT-NUMBER-----1		
DEVTYPE-----X'3010200F'	PHYRECS/TRK-----	12 HI-U-RBA-----737280
EXTENT-TYPE-----X'40'		
VOLFLAG-----PRIME	TRACKS/CA-----	15
EXTENTS:		
LOW-CCHH-----X'040F0000'	LOW-RBA-----	0 TRACKS-----15
HIGH-CCHH-----X'040F000E'	HIGH-RBA-----	737279
VOLUME		
VOLSER-----JIAB10	PHYREC-SIZE-----	4096 HI-A-RBA-----2211840
EXTENT-NUMBER-----1		
DEVTYPE-----X'3010200F'	PHYRECS/TRK-----	12 HI-U-RBA-----2211840
EXTENT-TYPE-----X'40'		
VOLFLAG-----PRIME	TRACKS/CA-----	15
EXTENTS:		
LOW-CCHH-----X'00270000'	LOW-RBA-----	737280 TRACKS-----30
HIGH-CCHH-----X'0028000E'	HIGH-RBA-----	2211839

In the example, the data set for DB9AU.DSNDBD.DSNDB04.TEST1TWO.I0001.A001 table space is multi-volume. Although LISTCAT shows spanning volumes, in previous ISMF examples they also span storage groups because of the Extend option.

Another option is the use of a overflow storage group. The concept is similar to the Extend Storage Group, but is used for data set allocations to a new storage group, rather than extending an already created data set to a new storage group. Overflow storage groups can also be used as Extend Storage Groups.

To illustrate overflow storage groups, we use two storage groups DB9ATST3 and DB9AT3OV. Both are mod 3 devices and are empty before allocation.

Only page 1 is shown in Figure 5-40, because page 2 contains the default values as seen in the example.

POOL STORAGE GROUP ALTER

Page 1 of 2

Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS

Storage Group Name : DB9ATST3

To ALTER Storage Group, Specify:

Description ==>

==>

Auto Migrate . . N (Y, N, I or P)

Auto Backup . . N (Y or N)

Auto Dump . . . N (Y or N)

Overflow . . . . N (Y or N)

Migrate Sys/Sys Group Name . .

Backup Sys/Sys Group Name . .

Dump Sys/Sys Group Name . . .

Extend SG Name . . . . .

Copy Pool Backup SG Name . . .

Dump Class . . .

Dump Class . . .

Dump Class . . .

(1 to 8 characters)

Dump Class . . .

Dump Class . . .

ALTER

SMS Storage Group Status . . . N (Y or N)

POOL STORAGE GROUP ALTER

Page 1 of 2

Command ==>

SCDS Name . . . . . : SYS1.SMS.SCDS

Storage Group Name : DB9AT3OV

To ALTER Storage Group, Specify:

Description ==>

==>

Auto Migrate . . N (Y, N, I or P)

Auto Backup . . N (Y or N)

Auto Dump . . . N (Y or N)

Overflow . . . . Y (Y or N)

Migrate Sys/Sys Group Name . .

Backup Sys/Sys Group Name . .

Dump Sys/Sys Group Name . . .

Extend SG Name . . . . .

Copy Pool Backup SG Name . . .

Dump Class . . .

Dump Class . . .

Dump Class . . .

(1 to 8 characters)

Dump Class . . .

Dump Class . . .

ALTER

SMS Storage Group Status . . . N (Y or N)

Figure 5-40 Storage Group panels for DB9ATST3 and DB9AT3OV

282 DB2 9 for z/OS and Storage Management



Storage Group DB9ATST3 has Overflow set to N. Figure 5-41 shows LISTVOL for DB9ATST3.

VOLUME LIST							
Command ==>							
Enter Line Operators below:							
LINE	VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
OPERATOR	SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
---(1)---	-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
	JIAC01	1079272K	39	1692228K	0	1079051K	2

Figure 5-41 LISTVOL for Storage Group DB9ATST3

DB9AT3OV has Overflow set to Y making it eligible as an overflow storage group. See Figure 5-42.

VOLUME LIST							
Command ==>							
Enter Line Operators below:							
LINE	VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
OPERATOR	SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
---(1)---	-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
	JIAD01	1909311K	69	862189K	0	1909090K	2

Figure 5-42 LISTVOL for Storage Group DB9AT3OV

We can see from the storage group ACS routine that TEST3 data sets can be allocated to two separate storage groups. See Example 5-22.

Example 5-22 Storage Class and storage group ACS routines for TEST3 data sets

#### PROC STORCLAS

```
FILTLIST DB9ATST3 INCLUDE(DB9AU.DSNDB%.*.TEST3*.**)
  IF &DSN = &DB9ATST3
  THEN DO
    SET &STORCLAS = 'DB9ATST3'
  EXIT
  END
```

#### PROC STORGRP

```
SELECT
  WHEN (&STORCLAS = 'DB9ATST3')
  SET &STORGRP = 'DB9ATST3','DB9AT3OV'
END
END
```

If Overflow was not specified in the ISMF Storage Group panel for DB9AT3OV, allocation would be eligible to either storage group, not only one. SMS would have picked a volume in the best of the combined storage group volumes at the time of allocation. It would not have mattered that DB9ATST3 is first in the list in the ACS routine.

Four table spaces are created. See Example 5-23. The first three table spaces are 1,000 cylinders each, the fourth table space is 1 cylinder. The order of creation is table space TEST3ONE, TEST3TWO, TEST3THR, and TEST3FOR.

*Example 5-23 ISPF 3.4 of the four data sets created*

Command - Enter "/" to select action	Message	Volume
DB9AU.DSNDBD.DSNDB04.TEST3FOR.I0001.A001		JIAC01
DB9AU.DSNDBD.DSNDB04.TEST3ONE.I0001.A001		JIAC01
<b>DB9AU.DSNDBD.DSNDB04.TEST3THR.I0001.A001</b>		<b>JIAD01</b>
DB9AU.DSNDBD.DSNDB04.TEST3TWO.I0001.A001		JIAC01

Command - Enter "/" to select action	Tracks	%Used	XT
DB9AU.DSNDBD.DSNDB04.TEST3FOR.I0001.A001	15	?	1
DB9AU.DSNDBD.DSNDB04.TEST3ONE.I0001.A001	15000	?	1
<b>DB9AU.DSNDBD.DSNDB04.TEST3THR.I0001.A001</b>	<b>15000</b>	<b>?</b>	<b>1</b>
DB9AU.DSNDBD.DSNDB04.TEST3TWO.I0001.A001	15000	?	1

The first, second, and fourth data sets were created on volume JIAC01, which is part of storage group DB9ATST3. The third data set DB9AU.DSNDBD.DSNDB04.TEST3THR.I0001.A001 was created on volume JIAD01, which is part of the overflow storage group, DB9AT3OV.

The HIGH threshold for both storage groups was 85%.

Example 5-24 shows that volume JIAC01 is only 39% free, or 61% used. This is well below the 85% HIGH value. If the third table space, TEST3THR, which was 1,000 cylinders was allocated to this volume, it would have exceeded the HIGH value. As such, SMS chose to allocate the data set on the overflow volume instead because it was an empty volume and would not have the same 85% threshold problem. The fourth data set allocated was TEST3FOR with 1 cylinder. Even though the third data set allocated to the overflow storage group, the fourth data set was small enough as to not exceed the 85% HIGH threshold for storage group DB9ATST3.

*Example 5-24 ISMF volume panel for JIAC01*

VOLUME LIST							
Command ==>							
Scroll ==> PAGE							
Entries 1-1 of 1							
Data Columns 3-8 of 43							
Enter Line Operators below:							
LINE	VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
OPERATOR	SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
---(1)---	-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
	<b>JIAC01</b>	1078442K	<b>39</b>	1693058K	0	1078221K	2

Another scenario is to allocate the data set using multitiered storage groups. An ISMF Storage Class panel entry enables the multitiered storage groups.

Assume there are three storage groups: SG1, SG2, and SG3.

If the storage group ACS routine states SET &STORGRP = 'SG1', 'SG2', 'SG3' and no overflow or Extend Storage Groups are defined, the data set will be allocated to any of the three storage groups based on the SMS placement algorithm.

Note what happens when multitiered storage groups are enabled for this statement:

- ▶ SMS selects volumes from SG1 before SG2 or SG3.
- ▶ If all enabled volumes in SG1 are over the HIGH threshold, SMS selects from SG2.
- ▶ If all enabled volumes in SG2 are over the HIGH threshold, SMS selects from SG3.
- ▶ If all volumes are over the HIGH threshold, SMS selects from the quiesced volumes in the same order.

We have created a Storage Class with Multi-tiered SGs enabled. See Figure 5-43.

```
STORAGE CLASS DISPLAY                               Page 2 of 2
Command ==>

CDS Name      . . . . . : SYS1.SMS.SCDS
Storage Class Name : DB9ATST4

Guaranteed Space . . . . . : NO
Guaranteed Synchronous Write . . : NO
Multi-Tiered SGs . . . . . : YES
Parallel Access Volume Capability : NOPREF
Cache Set Name . . . . . :
CF Direct Weight . . . . . :
CF Sequential Weight . . . . . :
Lock Set Name . . . . . :
```

*Figure 5-43 Storage Class DB9ATST4 with Multi-Tiered SGs (storage groups) enabled*

Along with Storage Class DB9ATST4, two storage groups are created DB9AT4T1 and DB9AT4T2 with a HIGH value of 30% each.

Figure 5-44 shows DB9AT4T1.

POOL STORAGE GROUP ALTER		Page 1 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
Storage Group Name : DB9AT4T1		
To ALTER Storage Group, Specify:		
Description ==>		
==>		
Auto Migrate . . N	(Y, N, I or P)	Migrate Sys/Sys Group Name . .
Auto Backup . . N	(Y or N)	Backup Sys/Sys Group Name . .
Auto Dump . . . N	(Y or N)	Dump Sys/Sys Group Name . . .
Overflow . . . . N	(Y or N)	Extend SG Name . . . . .
		Copy Pool Backup SG Name . . .
Dump Class . . .		(1 to 8 characters)
Dump Class . . .		Dump Class . . .
Dump Class . . .		Dump Class . . .
ALTER SMS Storage Group Status . . . N (Y or N)		
POOL STORAGE GROUP ALTER		Page 2 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
Storage Group Name : DB9AT4T1		
To ALTER Storage Group, Specify:		
<b>Allocation/migration Threshold :</b>		
High . . 30	(1-99)	Low . . 1 (0-99)
Alloc/Migr Threshold Track-Managed: High . . 85	(1-99)	Low . . 1 (0-99)
Guaranteed Backup Frequency . . . . .	(1 to 9999 or NOLIMIT)	
BreakPointValue . . . . .	(0-65520 or blank)	

Figure 5-44 Storage Group DB9AT4T1

Figure 5-45 shows DB9AT4T2.

POOL STORAGE GROUP ALTER		Page 1 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
Storage Group Name : DB9AT4T2		
To ALTER Storage Group, Specify:		
Description ==>		
==>		
Auto Migrate . . N	(Y, N, I or P)	Migrate Sys/Sys Group Name . .
Auto Backup . . N	(Y or N)	Backup Sys/Sys Group Name . .
Auto Dump . . . N	(Y or N)	Dump Sys/Sys Group Name . . .
Overflow . . . . N	(Y or N)	Extend SG Name . . . . .
		Copy Pool Backup SG Name . . .
Dump Class . . .		(1 to 8 characters)
Dump Class . . .		Dump Class . . .
Dump Class . . .		Dump Class . . .
ALTER SMS Storage Group Status . . . N (Y or N)		
POOL STORAGE GROUP ALTER		Page 2 of 2
Command ==>		
SCDS Name . . . . . : SYS1.SMS.SCDS		
Storage Group Name : DB9AT4T2		
To ALTER Storage Group, Specify:		
<b>Allocation/migration Threshold :</b> High . . 30 (1-99) Low . . 1 (0-99)		
Alloc/Migr Threshold Track-Managed: High . . 85 (1-99) Low . . 1 (0-99)		
Guaranteed Backup Frequency . . . . . (1 to 9999 or NOLIMIT)		
BreakPointValue . . . . . (0-65520 or blank)		

Figure 5-45 Storage Group DB9AT4T2

Each storage group is an empty Model 3 device before allocation. See Figure 5-46.

VOLUME LIST							
Command ==>							
Enter Line Operators below:							
LINE	VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
OPERATOR	SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
---(1)---	-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
	JIAE12	1909311K	69	862189K	0	1909090K	2
VOLUME LIST							
Command ==>							
Enter Line Operators below:							
LINE	VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
OPERATOR	SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
---(1)---	-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
	JIAF01	1909311K	69	862189K	0	1909090K	2

Figure 5-46 ISMF LISTVOL display for JIAE12 (DB9AT4T1) and JIAF01 (DB9AT4T2)

Example 5-25 shows the Storage Class and Group ACS routines for multitiered storage groups.

Example 5-25 Storage Class and Group ACS routines for multitiered storage groups

```

PROC STORCLAS
FILTLIST DB9ATST4 INCLUDE(DB9AU.DSNDB%.*.TEST4*.**)
  IF &DSN = &DB9ATST4
  THEN DO
    SET &STORCLAS = 'DB9ATST4'
  EXIT
  END

PROC STORGRP
SELECT
  WHEN (&STORCLAS = 'DB9ATST4')
  SET &STORGRP = 'DB9AT4T1','DB9AT4T2'
END
END

```

Two table spaces were created with 1,000 cylinders each, as shown in Figure 5-47.

Command - Enter "/" to select action	Message	Volume	
-----			
DB9AU.DSNDBD.DSNDB04.TEST4ONE.I0001.A001		JIAE12	
DB9AU.DSNDBD.DSNDB04.TEST4TWO.I0001.A001		JIAF01	
Command - Enter "/" to select action	Tracks	%Used	XT
-----			
DB9AU.DSNDBD.DSNDB04.TEST4ONE.I0001.A001	15000	?	1
DB9AU.DSNDBD.DSNDB04.TEST4TWO.I0001.A001	15000	?	1

Figure 5-47 ISPF 3.4 display of data sets and volumes for multitiered storage groups

The first table space, TEST4ONE, is allocated to JIAE12 (storage group DB9AT4T1); the second table space TEST4TWO is allocated to JIAF01 (storage group DB9AT4T2). Both storage groups have a HIGH threshold of 30%. Because the volumes are mod 3s, allocating 1,000 cylinders for each table space occupies 31% of each volume. SMS reviewed the 1,000 cylinder request for best placement and determined that both storage groups would exceed the 30% HIGH threshold. As such, TEST4ONE was allocated to the first storage group (DB9AT4T1). When the second allocation for TEST4TWO was requested, the first storage group exceed the HIGH threshold, therefore the data set was allocated to the second storage group (DB9AT4T2).

## 5.2.5 Separation Profile

The Separation Profile is part of the SMS base configuration and allows data sets during allocation to be separated by physical control units. z/OS 1.11 allows for the separation by volume and extent pool also, and not only physical control unit.

You can use the Separation Profile to make sure the duplex pair of the active log and boot strap data sets do not reside on the same logical volume. Additional steps must be taken to make sure they do not reside on the same physical volume either, and if possible separated by physical control unit.

To use the Separation Profile, select option **8** from ISMF:

8 Control Data Set - Specify System Names and Default Criteria

The initial panel for Control Data Set opens (Figure 5-48).

```
CDS APPLICATION SELECTION
Command ==>

To Perform Control Data Set Operations, Specify:
  CDS Name . . 'SYS1.SMS.SCDS'
                                     (1 to 44 Character Data Set Name or 'Active')

Select one of the following Options:
  3 1. Display      - Display the Base Configuration
    2. Define       - Define the Base Configuration
    3. Alter        - Alter the Base Configuration
    4. Validate     - Validate the SCDS
    5. Activate     - Activate the CDS
    6. Cache Display - Display CF Cache Structure Names for all CF Cache Sets
    7. Cache Update - Define/Alter/Delete CF Cache Sets
    8. Lock Display  - Display CF Lock Structure Names for all CF Lock Sets
    9. Lock Update  - Define/Alter/Delete CF Lock Sets
If CACHE Display is chosen, Enter CF Cache Set Name . . *
If LOCK Display is chosen, Enter CF Lock Set Name . . . *
                                     (1 to 8 character CF cache set name or * for all)
```

*Figure 5-48 Alter of the base configuration*

Select option **3** to Alter the base configuration.

Although both pages are illustrated in Figure 5-49 on page 291, the Separation Profile is on page 1 only.



```

SCDS BASE ALTER                                Page 1 of 2
Command ==>

SCDS Name . : SYS1.SMS.SCDS
SCDS Status : VALID

To ALTER SCDS Base, Specify:

Description ==> BASE SMS CONFIG FOR OE
              ==>

Default Management Class . . . . . (1 to 8 characters)
Default Unit . . . . . (esoteric or generic device name)
Default Device Geometry
  Bytes/Track . . . . . 56664 (1-999999)
  Tracks/Cylinder . . . . . 15 (1-999999)
DS Separation Profile (Data Set Name)
  ==> 'MHLRES4.DATASET.SEP.PROFILE'

SCDS BASE ALTER                                Page 2 of 2
Command ==>

SCDS Name . : SYS1.SMS.SCDS
SCDS Status : VALID

Specify one of the following options . . . (1 Add, 2 Delete, 3 Rename)

  Specify System Name . . . . . or Sys Group Name . .
  New System/Sys Group Name . . (For option 3, Rename)

System: SC63      SC64      SC65      SC70

Sysgrp:

```

Figure 5-49 SCDS base Alter panel

The Separation Profile points to MHLRES4.DATASET.SEP.PROFILE data set. This data set contains the separation rules.

The Separation Profile in this example requires volume separation for active log data sets DB9AU.LOGCOPY1.DS05 and its software dual-copy pair DB9AU.LOGCOPY2.DS05 (Example 5-26).

*Example 5-26 Separation Profile data set entry*

---

```

SEP(VOL) -
TYPE(REQ) -
DSNLIST(DB9AU.LOGCOPY1.DS05,DB9AU.LOGCOPY2.DS05);

```

---

As shown in Example 5-27, both data sets were created with 50 MB. Guaranteed Space was not enabled for either data set. Both data sets were allocated onto different volumes because of the Separation Profile.

*Example 5-27 IDCAMS DEFINE of the DS05 log data sets*

---

```

DEFINE CLUSTER -
  ( NAME (DB9AU.LOGCOPY1.DS05) -
    MB(50,0) -
    LINEAR )
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX0G IS 0
IDC0512I NAME GENERATED-(D) DB9AU.LOGCOPY1.DS05.DATA
IDC0181I STORAGECLASS USED IS DB9A
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

DEFINE CLUSTER -
  ( NAME (DB9AU.LOGCOPY2.DS05) -
    MB(50,0) -
    LINEAR )
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX0H IS 0
IDC0512I NAME GENERATED-(D) DB9AU.LOGCOPY2.DS05.DATA
IDC0181I STORAGECLASS USED IS DB9A
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0

```

---

Guaranteed Space can be used in combination with the Separation Profile. Example 5-28 shows both data sets using a Storage Class that has Guaranteed Space enabled and pointing to the same volume. The LOGCOPY1 data set is allocated to volume SBOX1Y as requested, however the LOGCOPY2 data sets fails the allocation because of the Separation Profile as seen in message IGD17279I.

*Example 5-28 DEFINE using Guaranteed Space, point to the same volume*

---

```

DEFINE CLUSTER -
  ( NAME (DB9AU.LOGCOPY1.DS05) -
    STORAGECLASS(DB9AGS) -
    VOLUMES(SBOX1Y) -
    MB(50,0) -
    LINEAR )
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IDC0508I DATA ALLOCATION STATUS FOR VOLUME SBOX1Y IS 0
IDC0512I NAME GENERATED-(D) DB9AU.LOGCOPY1.DS05.DATA
IDC0181I STORAGECLASS USED IS DB9AGS
IDC0181I MANAGEMENTCLASS USED IS MCDB22
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

DEFINE CLUSTER -
  ( NAME (DB9AU.LOGCOPY2.DS05) -
    STORAGECLASS(DB9AGS) -
    VOLUMES(SBOX1Y) -
    MB(50,0) -

```

```

        LINEAR )
IGD01009I MC ACS GETS CONTROL &ACSENVIR=ALLOC
IGD17217I UNABLE TO USE VOLUME SBOX1Y FOR GUARANTEED SPACE REQUEST
FOR DATA SET DB9AU.LOGCOPY2.DS05
IGD17807I THERE ARE (14) CANDIDATE VOLUMES OF WHICH (12) ARE ENABLED OR QUIESCED
IGD17290I THERE WERE 1 CANDIDATE STORAGE GROUPS OF WHICH THE FIRST 1
WERE ELIGIBLE FOR VOLUME SELECTION.
THE CANDIDATE STORAGE GROUPS WERE:DB9A
IGD17279I 2 VOLUMES WERE REJECTED BECAUSE THEY WERE NOT ONLINE
IGD17279I 2 VOLUMES WERE REJECTED BECAUSE THE UCB WAS NOT AVAILABLE
IGD17279I 1 VOLUMES WERE REJECTED BECAUSE THEY DID NOT MEET REQUIRED SEPARATION
IGD17219I UNABLE TO CONTINUE DEFINE OF DATA SET
DB9AU.LOGCOPY2.DS05
IDC3003I FUNCTION TERMINATED. CONDITION CODE IS 12

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 12

```

---

## 5.3 Considerations for DB2 data set types

In this section, we look at the following considerations when implementing SMS-managed storage for each of the DB2 data set types mentioned in 5.1.1, “DB2 data set considerations” on page 238:

- ▶ Active logs and BSDS
- ▶ DB2 catalog and directory
- ▶ DB2 user table spaces and indexes
- ▶ Archive log data sets on disk
- ▶ Image copy data sets on disk
- ▶ DB2 work file data sets (DSNDB07 or equivalent)

### 5.3.1 Active logs and BSDS

Active log and boot strap data sets must be separated from all other volumes to provide the mechanism for fast recovery. The following constructs must be set up to SMS-manage all of the DB2 active log and boot strap data sets:

- ▶ Data Class
  - Enable EF if striping the active log data sets. Otherwise, nothing specific is required in the Data Class.
- ▶ Storage Class
  - If striping the active log data sets, set an appropriate SDR value, typically 2 or 4.
  - Either assign Guaranteed Space to the active log and boot strap data sets or use the Separation Profile.
- ▶ Management Class
  - Assign the data sets to a class with no actions to be taken, such as MCNOACT.
- ▶ Storage Group
  - Define a storage group specifically for the active log and boot strap data sets. This storage group must be for that subsystem or Data Sharing group only and not shared with others.

- The software-managed dual-copy pairs must be on separate disk controllers if at all possible, otherwise on separate physical volumes, preferably behind separate logical subsystems/logical control units (LSS/LCU). For the separation, either the Separation Profile is used or the Storage Class attribute of Guaranteed Space is enabled.
- Set the value for HIGH to 99.
- Retain all other default values in the ISMF panels for the storage group.
- Assign a Copy Pool backup storage group if using FlashCopy.
- ▶ ACS routines
  - The active log and boot strap data sets have specific name qualifiers, with the names generally containing LOGCOPY% and BSDS\*. Use filter lists with the name pattern or specify a high-level qualifier in combination of LOGCOPY% and BSDS\* to assign the active log and BSDS data sets.

### 5.3.2 DB2 catalog and directory

DB2 10 for z/OS requires the DB2 catalog and directory to be SMS-managed with EF/EA enabled. The following constructs must be set up to SMS manage all of the DB2 catalog and directory data sets:

- ▶ Data Class
  - Enable EF and EA.
  - Provide for a dynamic volume count. The amount is installation-specific, but generally two or three volumes is sufficient. In DB2 10 for z/OS, this requirement is removed when DB2 catalog and directory data sets are converted to DB2 managed.
  - Set Extent Constraint Removal to YES
- ▶ Storage Class
  - Nothing specific is required, although you must assign a class so that the data sets can be SMS-managed. Guaranteed Space is not required.
- ▶ Management Class
  - Assign the data sets to a class with no actions to be taken, such as MCNOACT.
- ▶ Storage Group
  - Define a storage group specifically for the DB2 catalog and directory. This storage group must be for that subsystem or Data Sharing group only, and not shared with others.
  - Retain default values in the ISMF panels for the storage group.
  - Assign a Copy Pool backup storage group if using FlashCopy.
- ▶ ACS routines
  - The third qualifier is DSNDB01 for the directory and DSNDB06 for the catalog. Use filter lists with the name pattern or specify a high-level qualifier in combination of the third qualifier to assign the Catalog and Directory data sets.

### 5.3.3 DB2 user table spaces and indexes

DB2 user table spaces and indexes can be SMS-managed in a variety of ways. In this section, we are assuming that the DB2 user table spaces and indexes are DB2-managed, and not user-managed with IDCAMS DEFINE for the objects. User-managed data sets can follow the DB2 catalog and directory strategy without the volume and control unit separation.

The following constructs must be set up to SMS-manage all of the DB2 table spaces and indexes:

- ▶ **Data Class**
  - Enable EF and EA for ease of data set allocation.
  - Set Space Constraint Relief to YES.
  - If space availability is a periodic problem, set a reasonable value for Reduce Space Up To (%), such as 10%.
  - Set Extent Constraint Removal to YES.
- ▶ **Storage Class**
  - If striping any DB2 objects, set an appropriate SDR value.
  - Assign a multitiered storage group, if used.
- ▶ **Management Class**
  - Assign the data sets to a class with no actions to be taken, such as MCNOACT. For non-production environments, consider using release of unused space and migration of objects if desired.
- ▶ **Storage Group**
  - Define a storage group specifically for the DB2 user table spaces and indexes. This storage group must be for that subsystem or Data Sharing group only and not shared with others.
  - If migrating data sets, set the auto migration flag.
  - If migrating data sets, set a reasonable value for LOW.
  - Set a reasonable value for HIGH, consider starting at 85%, slightly higher for larger emulated disk models. If migration is what you want, consider setting HIGH at a lower level to allow for the migration.
  - Determine the use of overflow and Extend Storage Groups.
  - Retain default values in the ISMF panels for the Storage Group.
  - Assign a Copy Pool backup Storage Group if using FlashCopy.
  - Use the Separation Profile for volumes starting with z/OS 1.11 to avoid any known hot spots.
  - If using EAV volumes, assign a specific BPV (Break Point Value) is different than the one in PARMLIB.
- ▶ **ACS routines**
  - Use filter lists with the name pattern or specify a high-level qualifier in combination of the remainder of the DB2 data set name pattern to assign DB2 user table spaces and indexes.

### 5.3.4 Archive log data sets on disk

Archive log data sets must be separated from all other volumes to provide the mechanism for fast recovery. Allocating the archive log data sets to disk, even if later migrated by DFSMSHsm, is preferable over tape because it allows the data sets to be shared and recovery jobs can be executed in parallel.

The following constructs must be set up to SMS-manage all of the DB2 archive log data sets:

- ▶ Data Class
  - LARGE or EF must be used if the data set is greater than 4,369 cylinders. LARGE and EF are mutually exclusive. If the archive log data sets are striped, EF must be used.
  - Set `Compaction=YES` if compressing the archive log data sets.
- ▶ Storage Class
  - If striping the archive log data sets, set an appropriate SDR value.
- ▶ Management Class
  - If migrating the archive log data sets, assign a Management Class to migrate to ML1, then eventually to ML2 if you want, or directly to ML2.
- ▶ Storage Group
  - Define a storage group specifically for the archive log data sets. This storage group should be for that subsystem or Data Sharing group only and not share with others.
  - Verify that if the software-created dual copy pair resides on disk or tape, they do not reside on the same physical volume, or there is another copy of the data sets in case of physical failure.
  - If migrating data sets, set the auto migration flag.
  - If migrating data sets, set a reasonable value for LOW. Consider a value of 0 for archive log data sets.
  - Set a reasonable value for HIGH. If migration is what you want, consider setting HIGH at a lower level to allow for the migration.
  - Retain all other default values in the ISMF panels for the storage group.
  - If using EAV volumes, assign a specific Break Point Value (BPV) different from the one in PARMLIB.
- ▶ ACS routines
  - The ACS routines for archive log data sets must be based on the ZPARM values for ARCPFX1 and ARCPFX2 if both data sets are allocated to disk.

### 5.3.5 Image copy data sets on disk

Image copy data sets on disk can provide faster recovery when compared to the data sets stacked on tape. Image copy data sets on tape can result in slower recovery because the requests might have to be single threaded, waiting for a multiple data set tape. The following constructs must be set up to SMS-manage all of the DB2 image copy data sets:

- ▶ Data Class
  - LARGE or EF must be used If the data set is greater than 4,369 cylinders. LARGE and EF are mutually exclusive. If the image copy data sets are striped, EF must be used.
  - Volume Count and Dynamic Volume Count can be used if image copy data sets are requested as multi-volume.
  - Set Space Constraint Relief to YES
  - If space available is a periodic problem, set a reasonable value for Reduce Space Up To (%), such as 10%.
- ▶ Storage Class
  - If striping the image copy data sets, set an appropriate SDR value.

- ▶ **Management Class**
  - If migrating the image copy data sets, assign a Management Class to migrate to ML1, then eventually to ML2 if you want, or directly to ML2.
- ▶ **Storage Group**
  - Define a storage group specifically for the image copy data sets. This storage group must be for that subsystem or Data Sharing group only, and not shared with others.
  - Verify that if the software-created dual copy pair resides on disk or tape, they do not reside on the same physical volume, or there is another copy of the data sets in case of physical failure.
  - If migrating data sets, set the auto migration flag.
  - If migrating data sets, set a reasonable value for LOW. Consider a value of 0 for image copy data sets.
  - Set a reasonable value for HIGH. Remember, if migration is what you want, consider setting HIGH at a lower level to allow for the migration.
  - Retain all other default values in the ISMF panels for the storage group.
  - If using EAV volumes, assign a specific BPV (Break Point Value) different from the one in PARMLIB.
- ▶ **ACS routines**
  - The ACS routines for image copy data sets should be based on image copy data set name standards.

### 5.3.6 DB2 work file data sets (DSNDB07 or equivalent)

DB2 9 combines traditional work files and DGTT into work files. APARS PK70060 and PM02528 were introduced to provide flexibility to manage the DGTT files.

Work files used for sort operations can be managed by the user or DB2; when used for DGTT can only be DB2-managed. DGTTs can use only one work file table space; sort work files can use multiple work file table spaces. A commonality is that either type can be allocated as multi-volume.

How to SMS-manage the work files depends on the strategy in place for user-managed versus DB2-managed. The user-managed method follows the SMS rules for the DB2 catalog and directory; the DB2-managed method follows the rules that more aligned with DB2 user table spaces and indexes.

Neither volume separation nor migration is required. Other attributes, such as Guaranteed Space is not required. If enough PAVs are in place, the work file data sets can be added to an existing storage group, such as the DB2 user table spaces and indexes.

ACS routines must distinguish between user- and DB2-managed work files. One of the key differences is what happens during end-of-volume (EOV) processing. In this case, it is the Data Class that requires the most amount of analysis. User-managed work files must follow the EOV rules using VC and DVC; DB2-managed work files follow normal DB2 STOGROUP EOV processing and must not use VC, DVC, or both.

DB2-managed data sets do not use VC or DVC and they are ignored if specified in the Data Class. At EOV, DB2-managed data sets can be allocated one volume at a time to the DFSMSdfp maximum of 59 volumes rather than the artificial limit restricted by VC, DVC, or both.

## 5.4 Converting from non-SMS to SMS

Converting non-SMS-managed data sets to SMS-managed can be accomplished in several ways. Four possible approaches are as follows:

- ▶ Convert data sets in place by executing the DFSMSdss CONVERTV utility.
- ▶ Perform COPY, or DUMP and RESTORE tasks by using the DFSMSdss utility.
- ▶ Migrate the data sets using DFSMSHsm with commands HMIGRATE and HRECALL.
- ▶ Reallocate the data sets using REORG without REUSE.

The first option converts data sets in place on their original volumes and therefore requires those volumes be added to the SMS storage group. The remaining three options reallocate data sets to volumes that are already SMS-managed.

Generally the first option, using CONVERTV, is the fastest because no data movement is required. Prior to executing CONVERTV, the non-SMS-managed volumes have the following requirements:

- ▶ VTOC
- ▶ VTOC index
- ▶ VVDS, which is required for non-VSAM data sets also

Part of the CONVERTV process is to update the VVDS with the SMS class information. The VVDS must be large enough for the conversion process. Using IDCAMS to print the VVDS shows the SMS class entries.

Example 5-29 shows the output for the DB9AU.DSNDBC.DSNDB04.JOHNLAGR.I0001.A001 data set, which resides on volume SBOX1Y and its entry in the VVDS.

*Example 5-29 LISTCAT and PRINT of SMS-managed DB2 LDS*

---

```

CLUSTER ----- DB9AU.DSNDBC.DSNDB04.JOHNLAGR.I0001.A001
IN-CAT --- UCAT.DB9A
HISTORY
  DATASET-OWNER----PAOLOR9      CREATION-----2010.130
  RELEASE-----2              EXPIRATION-----0000.000
SMSDATA
  STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
  DATACLASS  -----DB2EFEA    LBACKUP ---0000.000.0000
VOLUME
  VOLSER-----SBOX1Y

//STEP1   EXEC   PGM=ADRDSSU,REGION=4M
//DASD    DD     UNIT=SYSALLDA,DISP=SHR,VOL=SER=SBOX1Y
//SYSPRINT DD    SYSOUT=*
//SYSIN   DD     *
  PRINT DATASET(SYS1.VVDS.VSBOX1Y) INDDNAME(DASD)
/*

4BC4E2D5 C4C2C44B C4E2D5C4 C2F0F44B *...Z.....DB9AU.DSNDBD.DSNDB04.*
F0F0F100 29C4C2F9 C1E44BC4 E2D5C4C2 *JOHNLAGR.I0001.A001..DB9AU.DSNDB*
D9C74BC9 F0F0F0F1 4BC1F0F0 F10009E4 *C.DSNDB04.JOHNLAGR.I0001.A001..U*
E2D5C4C2 C34BC4E2 D5C4C2F0 F44BD1D6 *CAT.DB9A.DB9AU.DSNDBC.DSNDB04.JO*
F100009E 21E0A000 00002000 0003E800 *HNLARG.I0001.A001.....Y.*
00FFFFFF FFFFFFFF FF000000 00880000 *..{...{...}....{.....h..*
00000000 0000000A FC800119 40200000 *.....*
00000000 04C4C2F9 C10007C4 C2F2C5C6 *.....DB9A..DB2EF*
00000000 00000000 00000000 00000000 *EA..MCDB22.....*
```

---



CONVERTV can run extremely fast compared to moving data. In our test, we executed CONVERTV for two full mod 3 volumes with four DB2 LDSs each. The entire conversion ran for less than one second. Another test was done for the same environment and same data sets using COPY to move the data sets from their non-SMS-managed volumes to SMS-managed volumes. The COPY process took 17 seconds compared to less than one second for CONVERTV. Execution times are highly dependent upon the number of volumes and data sets required to convert.

## Using an alternative to CONVERTV

Use an alternative to CONVERTV in the following situations:

- ▶ The volume has VTOC, VTOC index, or VVDS problems.
- ▶ The VVDS is not large enough to accommodate the conversion.
- ▶ The volume is full and cannot accommodate any additional increases for the VVDS conversion for a large number of data sets.

## Verifying data set is not SMS-managed

In our example, we chose that data sets with the following pattern to be non-SMS-managed:

DB9AU.DSNDB%.DSNDB04.NONSMS\*.\*\*

To verify that data sets are non-SMS-managed, we ran a test using ISMF option 7, which is located in the primary ISMF panel (Figure 4-1 on page 195):

## 7 Automatic Class Selection - Specify ACS Routines and Test Criteria

The ACS application selection opens (Figure 5-50).

```

ACS APPLICATION SELECTION
Command ==>

Select one of the following options:
1. Edit           - Edit ACS Routine source code
2. Translate      - Translate ACS Routines to ACS Object Form
3. Validate       - Validate ACS Routines Against Storage Constructs
4. Test          - Define/Alter Test Cases and Test ACS Routines
5. Display        - Display ACS Object Information
6. Delete        - Delete an ACS Object from a Source Control Data Set

If Display Option is Chosen, Specify:

CDS Name   . . 'SYS1.SMS.SCDS'
                                (1 to 44 Character Data Set Name or 'Active')

```

Figure 5-50 ISMF option 7 shows the ACS panel

To test the data set name pattern, select option **4**. You are then prompted for a library name and member. Select one of the following options:

- If this is a new test, select option **1**, and then enter a new member name.
- If this is an existing test case, select option **2** and then enter the existing member name.

The panel, shown in Figure 5-51 opens, where you enter a new member name.

```

ACS TEST SELECTION
Command ==>

Select one of the following Options:

    1. DEFINE      - Define an ACS Test Case
    2. ALTER       - Alter an ACS Test Case
    3. TEST        - Test ACS Routines

If DEFINE or ALTER Option is Chosen, Specify:

ACS Test Library . . JCL
ACS Test Member  . . ACSTESTD

```

Figure 5-51 IACS TEST SELECTION panel

Four pages of variations can be tested. In this case, we tested only the data set name on page 1, as shown in Figure 5-52.

```

ACS TEST CASE ALTER                                Page 1 of 4
Command ==>

ACS Test Library : DB2R2.JCL
ACS Test Member  . : ACSTESTD

To ALTER ACS Test Case, Specify:
Description ==>
Expected Result
DSN (DSN/Collection Name) . . DB9AU.DSNDBD.DSNDB04.NONSMS1.I0001.A001
MEMN (Object Name) . . . . .
Sysname . . . . . Xmode . . . . . Def_dataclas . .
Sysplex . . . . . ACSenvir . . . . . Def_mgmtclas . .
DD . . . . . Dataclas . . . . . Def_storclas . .
Dsorg . . . . . Mgmtclas . . . . . Dsntype . . . . .
Recorg . . . . . Storclas . . . . . If Ext . . . . .
Dstype . . . . . Storgrp . . . . . Seclabel . . . . .
Downer . . . . . Size . . . . . Space_Type . . . . .
Expdt . . . . . Maxsize . . . . . Second_Qty . . . . .
Retpd . . . . . Blksize . . . . .

```

Figure 5-52 Test the DSN option

Enter PF3 when done, then select option **3** for Test.

The next panel, shown in Figure 5-53 on page 301, has the library and member name, the SMS classes, and storage group that is to be tested. In this scenario, all ACS routines are to be tested.

```

TEST ACS ROUTINES
Command ==>

To Perform ACS Testing, Specify:

CDS Name . . . . . 'SYS1.SMS.SCDS'
                                   (1 to 44 Character Data Set Name or 'Active')
ACS Test Library . . JCL
ACS Test Member . . ACSTESTD (fully or partially specified or * for all
                               members)
Listing Data Set . . SMSTEST.OUTPUT
                               (1 to 44 Character Data Set Name or Blank)

Select which ACS Routines to Test:

DC ==> Y (Y/N)  SC ==> Y (Y/N)  MC ==> Y (Y/N)  SG ==> Y (Y/N)

```

Figure 5-53 Test ACS routines panel

Press Enter to display the results (Figure 5-54).

```

ACS TESTING RESULTS

CDS NAME       : SYS1.SMS.SCDS
ACS ROUTINE TYPES: DC SC MC SG
ACS TEST LIBRARY : DB2R2.JCL

ACS TEST
MEMBER          EXIT CODE  RESULTS
-----
DESCRIPTION:
EXPECTED RESULT:
ACSTESTD          0  DC = NULL VALUE ASSIGNED
                  0  SC = NULL VALUE ASSIGNED
NOTE: MC AND SG NOT EXECUTED WHEN ACS READ/WRITE VARIABLE STORCLAS = ''

ACS TESTING RC:  00

```

Figure 5-54 ACS routine test results

Figure 5-54 shows that no Data Class was assigned, but more important, no Storage Class was assigned and was set to null. This information means that data sets fitting the specified pattern are non-SMS-managed.

After execution, the option to print or delete the test output data set is available. The data set can be left as is.

Because we know that the Data Class is not invoked, we care only about the Storage Class ACS routine. We can find the location of the Storage Class by using ISMF option 7, same as the test. However, instead of option 4 for test, select option 5 to open the panel of Figure 5-55 on page 302.

**Note:** Make sure you are reviewing the correct CDS:

5. Display                    - Display ACS Object Information

ACS OBJECT DISPLAY						
Command ==>						
CDS Name : SYS1.SMS.SCDS						
ACS Rtn	Source Data Set	ACS	Member	Last Trans	Last Date	Last Time
Type	Routine	Translated from	Name	Userid	Translated	Translated
-----	-----	-----	-----	-----	-----	-----
DATACLAS	EVEBYE.JCL.CNTL		DCACTIVE	HAIMO	2010/05/03	21:16
MGMTCLAS	MHLRES4.JCL.CNTL		MCACTIVE	MHLRES2	2009/09/11	18:24
<b>STORCLAS</b>	<b>EVEBYE.JCL.CNTL</b>		<b>SCACTIVE</b>	HAIMO	2010/05/17	15:18
STORGRP	EVEBYE.JCL.CNTL		SGACTIVE	HAIMO	2010/05/17	15:18

Figure 5-55 ISMF option 7, then option 5: Display ACS Object Information

We know that the Storage Class ACS routine is housed in the EVEBYE.JCL.CNTL library, member SCACTIVE.

Browsing member SCACTIVE, we verify that the Storage Class sets our data set to null, which makes the data sets that match this pattern non-SMS-managed. See Example 5-30.

Example 5-30 Storage Class member SCACTIVE

```
PROC STORCLAS

FILTLIST DB9ANOT1 INCLUDE(DB9AU.DSNDB%.*.NONSMS*.**)

IF &DSN = &DB9ANOT1
  THEN DO
    SET &STORCLAS = ''
  EXIT
END
```

### 5.4.1 CONVERTV approach

For our test, two empty mod 3 devices were used with only the VTOC and VTOC index residing on volumes JI8118 and JI8119. The VVDS was automatically created by DFSMSdftp with a minimal amount of space after the first request for an LDS was received. Example 5-31 on page 303 shows the output of IEHLIST for volume JI8118.

*Example 5-31 IEHLIST for volumes JI8118*

---

```
//LIST29 EXEC PGM=IEHLIST
//DD1 DD VOL=SER=JI8118,DISP=SHR,UNIT=SYSALLDA
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LISTVTOC VOL=SYSALLDA=JI8118
/*
SYSTEMS SUPPORT UTILITIES---IEHLIST
DATE: 2010.137 TIME: 14.25.16
CONTENTS OF VTOC ON VOL JI8118 <THIS VOLUME IS NOT SMS MANAGED>
THERE IS A 1 LEVEL VTOC INDEX
DATA SETS ARE LISTED IN ALPHANUMERIC ORDER
-----DATA SET NAME----- CREATED DATE.EXP FILE TYPE SMS.IND EXT DS.VOL1
VOL.SEQ SECURITY
SYS1.VTOCIX.JI8118 2010.137 00.000 SEQUENTIAL 1 JI8118 1 NONE
THERE ARE 3300 EMPTY CYLINDERS PLUS 14 EMPTY TRACKS ON THIS VOLUME
THERE ARE 3300 EMPTY CYLINDERS PLUS 14 EMPTY TRACKS FROM THE TRACK-MANAGED SPACE
THERE ARE 23997 BLANK DSCBS IN THE VTOC ON THIS VOLUME
THERE ARE 1882 UNALLOCATED VIRS IN THE INDEX
```

---

We execute IEHLIST also for volume JI8119.

A DB2 STOGROUP is then created to include non-SMS-managed volumes JI8118 and JI8119, as shown in Example 5-32.

*Example 5-32 CREATE STOGROUP DDL for non-SMS-managed volumes JI8118 and JI8119*

---

```
CREATE STOGROUP NONSMS1
VOLUMES("JI8118","JI8119")
VCAT DB9AU;
```

---

Example 5-33 shows that each mod 3 volume has 3,300 cylinders free. DDL was executed to fill up all but 10 cylinders on each volume.

*Example 5-33 CREATE TABLESPACE to fill up both volumes, except for 10 cylinders on each*

---

Six table spaces were created with 1,000 cylinders each:

```
CREATE TABLESPACE NONSMS1
USING STOGROUP NONSMS1
PRIQTY 720000
SECQTY 72000
LOCKSIZE ANY
CLOSE NO
BUFFERPOOL BP0
FREEPAGE 0
PCTFREE 0;
```

Two table spaces were created with 290 cylinders each:

```
CREATE TABLESPACE NONSMS2
USING STOGROUP NONSMS1
PRIQTY 208800
SECQTY 72000
LOCKSIZE ANY
CLOSE NO
BUFFERPOOL BP0
FREEPAGE 0
PCTFREE 0;
```

---

Example 5-34 shows the output of IEHLIST of volume JI8118 after the table space allocations.

**Example 5-34 IEHLIST for non-SMS volume JI8118 after allocations**

---

```
//LIST29 EXEC PGM=IEHLIST
//DD1 DD VOL=SER=JI8118,DISP=SHR,UNIT=SYSALLDA
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LISTVTOC VOL=SYSALLDA=JI8118
/*
```

1 SYSTEMS SUPPORT UTILITIES---IEHLIST PAGE 1

-DATE: 2010.137 TIME: 14.30.49

CONTENTS OF VTOC ON VOL JI8119 <THIS VOLUME IS NOT SMS MANAGED>

THERE IS A 1 LEVEL VTOC INDEX

DATA SETS ARE LISTED IN ALPHANUMERIC ORDER

-----DATA SET NAME-----	CREATED	DATE.EXP	FILE TYPE	SMS.IND	EXT	DS.VOL1	VOL.SEQ	SECURITY
DB9AU.DSNDBD.DSNDB04.NONSMS4.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS5.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS6.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS8.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
SYS1.VTOCIX.JI8119	2010.137	00.000	SEQUENTIAL		1	JI8119	1	NONE
SYS1.VVDS.VJI8119	2010.137	00.000	VSAM		1	JI8119	1	PWD
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS ON THIS VOLUME								
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS FROM THE TRACK-MANAGED SPACE								
THERE ARE 23992 BLANK DSCBS IN THE VTOC ON THIS VOLUME								
THERE ARE 1882 UNALLOCATED VIRS IN THE INDEX								

---

Example 5-35 shows the output of IEHLIST of volume JI8119 after the table space allocations.

**Example 5-35 IEHLIST for non-SMS volume JI8119 after allocations**

---

```
//LIST29 EXEC PGM=IEHLIST
//DD1 DD VOL=SER=JI8119,DISP=SHR,UNIT=SYSALLDA
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LISTVTOC VOL=SYSALLDA=JI8119
/*
```

1 SYSTEMS SUPPORT UTILITIES---IEHLIST PAGE 1

-DATE: 2010.137 TIME: 14.30.49

CONTENTS OF VTOC ON VOL JI8119 <THIS VOLUME IS NOT SMS MANAGED>

THERE IS A 1 LEVEL VTOC INDEX

DATA SETS ARE LISTED IN ALPHANUMERIC ORDER

-----DATA SET NAME-----	CREATED	DATE.EXP	FILE TYPE	SMS.IND	EXT	DS.VOL1	VOL.SEQ	SECURITY
DB9AU.DSNDBD.DSNDB04.NONSMS4.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS5.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS6.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS8.I0001.A001	2010.137	1999.365	VSAM		1	JI8119	1	PWD
SYS1.VTOCIX.JI8119	2010.137	00.000	SEQUENTIAL		1	JI8119	1	NONE
SYS1.VVDS.VJI8119	2010.137	00.000	VSAM		1	JI8119	1	PWD
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS ON THIS VOLUME								
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS FROM THE TRACK-MANAGED SPACE								
THERE ARE 23992 BLANK DSCBS IN THE VTOC ON THIS VOLUME								
THERE ARE 1882 UNALLOCATED VIRS IN THE INDEX								

---

Example 5-36 lists the output of LISTCAT of one of the eight data sets that are created. There is no SMS class information.

*Example 5-36 LISTCAT of one of the eight data sets created, no SMS class information*

---

```
//DSNTDBL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LISTC ENT(DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001) ALL

LISTC ENT(DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001) ALL
CLUSTER ----- DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001
IN-CAT --- UCAT.DB9A
HISTORY
  DATASET-OWNER----PAOLOR9      CREATION-----2010.137
  RELEASE-----2             EXPIRATION-----0000.000
  EATTR----- (NULL)
  BWO STATUS----- (NULL)      BWO TIMESTAMP----- (NULL)
  BWO----- (NULL)
  PROTECTION-PSWD----- (NULL)  RACF----- (NO)
```

---

At this point, the ACS routines have been updated to allow for data sets meeting the following DB9AU.DSNDB%.\*.NONSMS\*.\*\* pattern to be SMS-managed, and volumes JI8118 and JI8119 are added to the storage group. CONVERTV is executed with the TEST option, as shown in Example 5-37.

*Example 5-37 CONVERTV with TEST option*

---

```
//STEP1 EXEC PGM=ADRDSSU,REGION=4M
//DASD0002 DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  CONVERTV SMS -
    DYNAM((JI8118),(JI8119)) -
  TEST
/*

1PAGE 0001      5695-DF175 DFSMSDSS V1R11.0 DATA SET SERVICES      2010.137 15:28
- CONVERTV SMS -
  DYNAM((JI8118),(JI8119)) -
  TEST
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'CONVERTV '
ADR109I (R/I)-RI01 (01), 2010.137 15:28:39 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR863I (R/I)-RI03 (01), THE TEST KEYWORD WAS SPECIFIED. NO DATA SETS OR VOLUMES WILL BE CONVERTED
ADRO16I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
OADR006I (001)-STEND(01), 2010.137 15:28:39 EXECUTION BEGINS
OADR860I (001)-KVSMS(01), PROCESSING BEGINS ON VOLUME JI8118
OADR873I (001)-KVSMS(01), VOLUME JI8118 IN STORAGE GROUP DB9A IS ELIGIBLE FOR CONVERSION TO SMS MANAGEMENT
OADR877I (001)-KVSMS(01), THE FOLLOWING DATA SETS ON VOLUME JI8118 WILL BE SUCCESSFULLY PROCESSED
0          DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001      CATALOG: UCAT.DB9A
              STORCLAS: DB9A                          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS2.I0001.A001      CATALOG: UCAT.DB9A
              STORCLAS: DB9A                          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS3.I0001.A001      CATALOG: UCAT.DB9A
              STORCLAS: DB9A                          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS7.I0001.A001      CATALOG: UCAT.DB9A
              STORCLAS: DB9A                          MGMTCLAS: MCDB22
OADR885I (001)-KVSMS(01), VOLUME JI8118 WILL BE SUCCESSFULLY CONVERTED TO SMS MANAGEMENT
OADR860I (001)-KVSMS(01), PROCESSING BEGINS ON VOLUME JI8119
OADR873I (001)-KVSMS(01), VOLUME JI8119 IN STORAGE GROUP DB9A IS ELIGIBLE FOR CONVERSION TO SMS MANAGEMENT
```

```

OADR877I (001)-KVSMS(01), THE FOLLOWING DATA SETS ON VOLUME JI8119 WILL BE SUCCESSFULLY PROCESSED
0          DB9AU.DSNDBC.DSNDB04.NONSMS4.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS5.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS6.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS8.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
OADR885I (001)-KVSMS(01), VOLUME JI8119 WILL BE SUCCESSFULLY CONVERTED TO SMS MANAGEMENT
1PAGE 0002      5695-DF175 DFSMSDSS V1R11.0 DATA SET SERVICES      2010.137 15:28
-ADR892I (001)-KVRPT(01), THE STATUS OF EACH VOLUME WILL BE AS FOLLOWS
0          VOLUME          FINAL STATUS          REASON FOR FAILURE
          -----          -
0          JI8118 - CONVERTED          SMS
          JI8119 - CONVERTED          SMS
OADR006I (001)-STEND(02), 2010.137 15:28:39 EXECUTION ENDS
OADR013I (001)-CLTSK(01), 2010.137 15:28:39 TASK COMPLETED WITH RETURN CODE 0000
OADR012I (SCH)-DSSU (01), 2010.137 15:28:39 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000

```

After the TEST option is verified, CONVERTV is executed. See Example 5-38. The PREPARE keyword is not used. (Specifying PREPARE prevents data sets from extending and new allocations from being made on the volume. However, users can still access the data on the volume from either the SMS system or a system that is sharing the volume.)

---

*Example 5-38 CONVERTV successfully completed*

---

```

//STEP1 EXEC PGM=ADRDSU,REGION=4M
//DASD0002 DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  CONVERTV SMS -
    DYNAM((JI8118),(JI8119))
/*

1PAGE 0001      5695-DF175 DFSMSDSS V1R11.0 DATA SET SERVICES      2010.137 15:30
- CONVERTV SMS -
  DYNAM((JI8118),(JI8119))
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'CONVERTV '
ADR109I (R/I)-RI01 (01), 2010.137 15:30:03 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
OADR006I (001)-STEND(01), 2010.137 15:30:03 EXECUTION BEGINS
OADR860I (001)-KVSMS(01), PROCESSING BEGINS ON VOLUME JI8118
OADR873I (001)-KVSMS(01), VOLUME JI8118 IN STORAGE GROUP DB9A IS ELIGIBLE FOR CONVERSION TO SMS MANAGEMENT
OADR877I (001)-KVSMS(01), THE FOLLOWING DATA SETS ON VOLUME JI8118 WERE SUCCESSFULLY PROCESSED
0          DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS2.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS3.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS7.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
OADR885I (001)-KVSMS(01), VOLUME JI8118 HAS BEEN SUCCESSFULLY CONVERTED TO SMS MANAGEMENT
OADR860I (001)-KVSMS(01), PROCESSING BEGINS ON VOLUME JI8119
OADR873I (001)-KVSMS(01), VOLUME JI8119 IN STORAGE GROUP DB9A IS ELIGIBLE FOR CONVERSION TO SMS MANAGEMENT
OADR877I (001)-KVSMS(01), THE FOLLOWING DATA SETS ON VOLUME JI8119 WERE SUCCESSFULLY PROCESSED
0          DB9AU.DSNDBC.DSNDB04.NONSMS4.I0001.A001          CATALOG: UCAT.DB9A
          STORCLAS: DB9A          MGMTCLAS: MCDB22
0          DB9AU.DSNDBC.DSNDB04.NONSMS5.I0001.A001          CATALOG: UCAT.DB9A

```



```

                                STORCLAS: DB9A                                MGMTCLAS: MCDB22
0                                DB9AU.DSNDBC.DSNDB04.NONSMS6.I0001.A001        CATALOG: UCAT.DB9A
                                STORCLAS: DB9A                                MGMTCLAS: MCDB22
0                                DB9AU.DSNDBC.DSNDB04.NONSMS8.I0001.A001        CATALOG: UCAT.DB9A
                                STORCLAS: DB9A                                MGMTCLAS: MCDB22
OADR885I (001)-KVSMS(01), VOLUME JI8119 HAS BEEN SUCCESSFULLY CONVERTED TO SMS MANAGEMENT
1PAGE 0002      5695-DF175  DFSMSDSS V1R11.0 DATA SET SERVICES      2010.137 15:30
-ADR892I (001)-KVRPT(01), THE STATUS OF EACH VOLUME IS AS FOLLOWS
0                                VOLUME          FINAL STATUS          REASON FOR FAILURE
                                -----
0                                JI8118 - CONVERTED      SMS
                                JI8119 - CONVERTED      SMS
OADR006I (001)-STEND(02), 2010.137 15:30:03 EXECUTION ENDS
OADR013I (001)-CLTSK(01), 2010.137 15:30:03 TASK COMPLETED WITH RETURN CODE 0000
OADR012I (SCH)-DSSU (01), 2010.137 15:30:03 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000

```

Volumes JI8118 and JI8119 are now SMS-managed, including all the data sets that reside on them.

See Example 5-39 for volume JI8118.

*Example 5-39 EHLIST for volume JI8118 after CONVERTV*

```

//LIST29 EXEC PGM=IEHLIST
//DD1 DD VOL=SER=JI8118,DISP=SHR,UNIT=SYSALLDA
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LISTVTOC VOL=SYSALLDA=JI8118
/*

1                                SYSTEMS SUPPORT UTILITIES---IEHLIST                                PAGE 1
-DATE: 2010.137 TIME: 15.30.32
      CONTENTS OF VTOC ON VOL JI8118 <THIS IS AN SMS MANAGED VOLUME>
      THERE IS A 1 LEVEL VTOC INDEX
      DATA SETS ARE LISTED IN ALPHANUMERIC ORDER
-----DATA SET NAME----- CREATED DATE.EXP FILE TYPE SMS.IND EXT DS.VOL1 VOL.SEQ SECURITY
DB9AU.DSNDBD.DSNDB04.NONSMS1.I0001.A001 2010.137 00.000 VSAM S 1 JI8118 1 PWD
DB9AU.DSNDBD.DSNDB04.NONSMS2.I0001.A001 2010.137 00.000 VSAM S 1 JI8118 1 PWD
DB9AU.DSNDBD.DSNDB04.NONSMS3.I0001.A001 2010.137 00.000 VSAM S 1 JI8118 1 PWD
DB9AU.DSNDBD.DSNDB04.NONSMS7.I0001.A001 2010.137 00.000 VSAM S 1 JI8118 1 PWD
SYS1.VTOCIX.JI8118 2010.137 00.000 SEQUENTIAL SU 1 JI8118 1 NONE
SYS1.VVDS.VJ8118 2010.137 00.000 VSAM S 1 JI8118 1 PWD
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS ON THIS VOLUME
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS FROM THE TRACK-MANAGED SPACE
THERE ARE 23992 BLANK DSCBS IN THE VTOC ON THIS VOLUME
THERE ARE 1882 UNALLOCATED VIRS IN THE INDEX

```

See Example 5-40 for volume JI8119.

*Example 5-40 IEHLIST for volume JI8119 after CONVERTV*

```

//LIST29 EXEC PGM=IEHLIST
//DD1 DD VOL=SER=JI8119,DISP=SHR,UNIT=SYSALLDA
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LISTVTOC VOL=SYSALLDA=JI8119
/*

1                                SYSTEMS SUPPORT UTILITIES---IEHLIST                                PAGE 1
-DATE: 2010.137 TIME: 15.30.41
      CONTENTS OF VTOC ON VOL JI8119 <THIS IS AN SMS MANAGED VOLUME>
      THERE IS A 1 LEVEL VTOC INDEX
      DATA SETS ARE LISTED IN ALPHANUMERIC ORDER

```

-----DATA SET NAME-----	CREATED	DATE.EXP	FILE TYPE	SMS.IND	EXT	DS.VOL1	VOL.SEQ	SECURITY
DB9AU.DSNDBD.DSNDB04.NONSMS4.I0001.A001	2010.137	00.000	VSAM	S	1	J18119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS5.I0001.A001	2010.137	00.000	VSAM	S	1	J18119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS6.I0001.A001	2010.137	00.000	VSAM	S	1	J18119	1	PWD
DB9AU.DSNDBD.DSNDB04.NONSMS8.I0001.A001	2010.137	00.000	VSAM	S	1	J18119	1	PWD
SYS1.VTOCIX.J18119	2010.137	00.000	SEQUENTIAL	SU	1	J18119	1	NONE
SYS1.VVDS.VJ18119	2010.137	00.000	VSAM	S	1	J18119	1	PWD
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS ON THIS VOLUME								
THERE ARE 10 EMPTY CYLINDERS PLUS 4 EMPTY TRACKS FROM THE TRACK-MANAGED SPACE								
THERE ARE 23992 BLANK DSCBS IN THE VTOC ON THIS VOLUME								
THERE ARE 1882 UNALLOCATED VIRS IN THE INDEX								

The LISTCAT output for the cluster portion of the data sets now contain the SMS classes information. Example 5-41 shows the LISTCAT output with the SMS classes.

*Example 5-41 LISTCAT example of converted data set showing SMS classes*

```
//DSNTDBL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
LISTC ENT(DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001) ALL
/*

LISTC ENT(DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001) ALL
CLUSTER ----- DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001
IN-CAT --- UCAT.DB9A
HISTORY
  DATASET-OWNER----PAOLOR9      CREATION-----2010.137
  RELEASE-----2              EXPIRATION-----0000.000
SMSDATA
  STORAGECLASS -----DB9A      MANAGEMENTCLASS---MCDB22
  DATACLASS -----(NULL)      LBACKUP ---0000.000.0000
  EATTR----- (NULL)
  BWO STATUS-----00000000      BWO TIMESTAMP----- (NULL)
  BWO----- (NULL)
```

## 5.4.2 COPY approach

One alternative to CONVERTV is to move data sets to SMS-managed volumes.

In our next example, DFSMSDss COPY is used to move the data sets after the ACS routines are altered to allow data sets that match the DB9AU.DSNDB%.\*.NONSMS%.\*\* pattern to be SMS-managed. See Example 5-42.

*Example 5-42 COPY from non-SMS volume J18118 and J18119 to SMS-managed volumes*

```
//STEP1 EXEC PGM=ADRDSSU,REGION=4M
//DASD0002 DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
COPY DATASET(INCLUDE(DB9AU.DSNDB%.DSNDB04.NONSMS%.**)) -
ALLDATA(*) -
ALLEXCP -
DELETE
/*

1PAGE 0001 5695-DF175 DFSMSDSS V1R11.0 DATA SET SERVICES 2010.137 15:04
- COPY DATASET(INCLUDE(DB9AU.DSNDB%.DSNDB04.NONSMS%.**)) -
ALLDATA(*) -
```

```

ALLEXCP -
DELETE
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'COPY '
ADR109I (R/I)-RI01 (01), 2010.137 15:04:13 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
OADR006I (001)-STEND(01), 2010.137 15:04:13 EXECUTION BEGINS
OADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
OADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS2.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS2.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS2.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
OADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS3.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS3.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS3.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
OADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS4.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS4.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS4.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
OADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS5.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS5.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS5.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
OADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS6.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS6.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS6.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
OADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS7.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS7.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS7.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
1PAGE 0002      5695-DF175  DFSMSDSS V1R11.0 DATA SET SERVICES      2010.137 15:04
-ADR711I (001)-NEWDS(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS8.I0001.A001 HAS BEEN ALLOCATED USING STORCLAS DB9A, NO
DATACLAS, AND
                                MGMTCLAS MCDB22
OADR806I (001)-TOMI (03), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS8.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
OADR431I (001)-XVSAM(01), DATA SET DB9AU.DSNDBC.DSNDB04.NONSMS8.I0001.A001 IN CATALOG UCAT.DB9A HAS BEEN DELETED
OADR801I (001)-DDDS (01), DATA SET FILTERING IS COMPLETE. 8 OF 8 DATA SETS WERE SELECTED: 0 FAILED SERIALIZATION AND 0
FAILED FOR
                                OTHER REASONS
OADR454I (001)-DDDS (01), THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED
0                                DB9AU.DSNDBC.DSNDB04.NONSMS1.I0001.A001
0                                DB9AU.DSNDBC.DSNDB04.NONSMS2.I0001.A001
0                                DB9AU.DSNDBC.DSNDB04.NONSMS3.I0001.A001
0                                DB9AU.DSNDBC.DSNDB04.NONSMS4.I0001.A001
0                                DB9AU.DSNDBC.DSNDB04.NONSMS5.I0001.A001
0                                DB9AU.DSNDBC.DSNDB04.NONSMS6.I0001.A001
0                                DB9AU.DSNDBC.DSNDB04.NONSMS7.I0001.A001
0                                DB9AU.DSNDBC.DSNDB04.NONSMS8.I0001.A001
OADR006I (001)-STEND(02), 2010.137 15:04:30 EXECUTION ENDS
OADR013I (001)-CLTSK(01), 2010.137 15:04:30 TASK COMPLETED WITH RETURN CODE 0000
OADR012I (SCH)-DSSU (01), 2010.137 15:04:30 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000

```

In this case, the COPY operation copied the data sets from non-SMS-managed volumes JI8118 and JI8119 to SMS-managed volumes. Volumes JI8118 and JI8119 are still non-SMS-managed, but all the data sets have been moved to SMS-managed volumes. The

only objects that remain on volumes JI8118 and JI8119 are the VTOC, VTOC index, and the VVDS. Volumes JI8118 and JI8119 are now ready to be used for other functions.

We have described the techniques for converting DB2 data to SMS. However, each customer has unique data sets and facilities to support their online environment. These differences have an effect on recommended storage management procedures. Database data has various space, performance, and availability requirements. Therefore, dividing database data into categories can help identify the required SMS services and implement a staged conversion to SMS.

### 5.4.3 Overview of the DB2 and SMS relationship

For the DB2 and SMS relationship to be successful, the data base administrator (DBA) must clearly specify the characteristics and requirements of the DB2 data sets. The storage administrator must also ensure they are satisfied in the physical implementation.

All types of DB2 data are important for successful operation of a DB2 environment. Great care must be taken in preparing DB2 for its conversion to SMS management.

Under most circumstances, an installation already has implemented SMS to some degree prior to considering the management of DB2; likely candidates are batch and TSO data sets. Therefore, an assumption is that sufficient skills exist within the storage administrator's area to provide the levels of support needed.

To gain experience with the various aspects of the DB2/SMS relationship, if possible, first convert a DB2 test system to DFSMS. The DB2 administrator and storage administrator must work closely together to test the environment. When satisfied with this scenario, develop a migration plan to convert DB2 data.

The technique and implementation sequence for converting a DB2 system to SMS varies according to each installation. However, the following topics provide a guideline:

- ▶ Advantages of SMS managing DB2 data
- ▶ SMS management goals
- ▶ Positioning for implementation
- ▶ Conversion processes
- ▶ DFSMS FIT
- ▶ NaviQuest

### 5.4.4 Advantages of SMS managing DB2 data

ACS routines can be designed so that SMS restricts the allocation of data sets in DB2 storage groups to production databases and selected system data sets. Only authorized users, such as the DB2 administrator or storage administrator can allocate data in these storage groups. They also have the authority to allocate data sets with critical performance and availability requirements to specific volumes. Dual-copy provides high availability for selected data sets that are not duplexed by the database management system. The use of fast-write and cache facilities can provide increased performance for databases and recovery data sets.

DFSMS/MVS enhances the backup and recovery utilities provided by the DB2 system as follows:

- ▶ DFSMSdss uses concurrent copy capability to create point-of-consistency backups.
- ▶ DFSMSHsm backs up system data sets and user database data that is less critical than production database data.

- ▶ DFSMSShsm carries out direct migration to migration level 2 for archived recovery data sets on disk storage.

Testing user databases can be migrated by DFSMSShsm through the storage hierarchy, based on database usage.

### 5.4.5 SMS management goals

The aims and goals for managing SMS differ for each installation, although there are areas where working practices are in common. These areas can be categorized as follows:

- ▶ Positioning for future enhancements to both DFSMS/MVS and DB2
- ▶ Improving the storage management of data:
  - Use of SMS to simplify JCL allocation.
  - Maintain support for disk and data storage growth without increasing staff levels.
  - Use of SMS to simplify data movement.
  - Improve disk storage efficiency by increasing space utilization through better use of allocation control.
  - Bring private disk volumes under centralized control.
  - Segregate production from other data.
  - Reduce disk storage requirements by migration of inactive data.

Improve the DB2 aspects of data management:

- ▶ Spread partitions for a given table/PI.
- ▶ Spread partitions of tables and indexes likely to be joined.
- ▶ Spread pieces of NPIs.
- ▶ Spread DB2 work files, and temporary data sets likely to be accessed in parallel.
- ▶ Make productive use of hardware such as RVA.
- ▶ Use physical striping of data.
- ▶ Avoid UCB contention.
- ▶ Use only what disk space is actually needed.

### 5.4.6 Positioning for implementation

This section describes prerequisite planning and service level agreements.

#### Prerequisite planning

For the DBA, a number of items must be considered as prerequisites for the process.

#### ***Categorize each data type into separate groups***

Consider the usage characteristics and service requirements for each data type:

- ▶ Response time performance
- ▶ Accessibility and availability operations
- ▶ Initial sizing of data sets and future growth
- ▶ Difference between production and user and testing data sets

#### ***DB2 STOGROUPs mapping to SMS storage groups***

To ensure consistency, be sure that DB2 STOGROUPs are converted to equivalent SMS storage groups.

### ***Identify DB2 data sets eligible for HSM management***

Decide for which groups of DB2 data DFSMSHsm will have the authority to migrate or backup. For example, production databases, active logs, system libraries, and BSDS are candidates for NO MIGRATION because of their critical status.

Set DSNZPARM to have DFSMSHsm automatically recall DB2 data sets during DB2 access. Set RECALL to YES. Set RECALLD, the maximum wait for DFSMSHsm to complete recreation of data sets on the primary disk, based on testing with typical user databases.

### ***Use of Guaranteed Space***

As part of the initial phase, the GUARANTEED SPACE option can be used to position data, particularly production table spaces, and active logs. When satisfied with the allocation of the data sets, remove this option, so future allocations can be under the sole control of SMS.

Guaranteed Space is for use only during the migration period (from DB2 managed to SMS-managed), which must be kept short to prevent failures on initial allocation and data set extension. Unlike non-SMS, SMS does not retry allocation on another volume if the requested space cannot be satisfied on the specified candidate volume.

Guaranteed Space is not useful unless the space requirements are relatively small and static.

### ***Ensure that all data sets are cataloged***

SMS requires that all data sets are cataloged in ICF catalogs, enabling the use of standard catalog search routines (after 1999, VSAM and CVOL catalogs were no longer). For further information see, *DFSMS/MVS Managing Catalogs*, SC26-4914.

### ***DB2 naming conventions***

Certain parts of table space names are generated by DB2. This process does not leave the DBA with much scope for a flexible naming convention. For further information about this subject see the following sections:

- ▶ 4.3.7, “Assigning SMS classes and storage groups for DB2 objects” on page 233
- ▶ 4.3.8, “SMS base configuration” on page 235

Ensure that the storage administrator is fully aware of any restrictions so ACS routines can be coded accordingly.

### ***DB2 recovery requirements***

For purposes of DB2 recovery, you must decide the degree of integrity required for active logs, image copies, and archive logs.

### ***Expiration of data sets***

Management Class expiration attributes must be synchronized with DB2's expiration information for old archived logs or obsolete image copies (generally not DB2 system or user objects):

- ▶ Expiration of archive logs must be consistent with the value of ARCRETN. The BSDS must be updated with the DB2 change log inventory utility to remove deleted archive logs.
- ▶ Expiration of archive logs must also be consistent with the expiration of image copies, which is described in “Deleting image copies and archive logs” on page 160.
- ▶ Expiration of any DB2 image copies requires running the MODIFY utility to update SYSCOPY.

### **Service level agreement (SLA)**

The SLA must be drawn up between the DBA and the storage administrator to include the following items (and the items in “Prerequisite planning” on page 311):

- ▶ The levels of service that are required by various data types
- ▶ Performance, accessibility, and availability characteristics
- ▶ The use of dedicated volumes
- ▶ The use of the GUARANTEED SPACE parameter
- ▶ The use of HSM management (automatic migration, recall, backup, space release, and data set expiration)
- ▶ Data set naming conventions

## **5.4.7 Conversion Process**

This topic covers aspects of planning and converting DB2 data.

### **Sequence**

To ensure minimum disruption to services, the following sequence is suggested for implementation:

1. Libraries and other DB2 system data sets
2. Archive logs and image copies
3. User and testing table spaces
4. Production table spaces
5. Active logs and BSDS

### **Methodology**

Methodology involves items described in this section.

#### ***Conversion window***

Decide when each type of data is available for conversion. During a normal processing cycle, certain data sets are deleted and reallocated, providing the opportunity for SMS management. Online data must be converted when those services are unavailable (for example, down time). Conversion is the most difficult to schedule, and requires precise planning.

#### ***Data movement***

Each disk device is either SMS-managed or it is not. A data set is considered SMS-managed in the following situations:

- ▶ It has a valid Storage Class.
- ▶ It resides on a volume in an SMS storage group, or has been migrated by DFSMSHsm.

Data sets can be converted with *movement* or *in place*:

- ▶ Converted with movement

This conversion is achieved by using a space management function such as DFSMSdss COPY, DFSMSdss DUMP/RESTORE or DFSMSHsm. This method is applicable if the data is application owned. However, consider the number of spare disk devices required, while this method is in progress. Also, consider using this approach if the disk devices being used are positioned with storage controls of varying performance (caching). An advantage with this method is allowing data to be allocated using volume thresholds that are set for each storage group, thus allowing space management to operate.

For table spaces, the DB2 utility REORG can be used to automatically convert with data movement if the table space is DB2 defined. If it user-defined, a IDCAMS DELETE/DEFINE CLUSTER must be executed between the REORG phases.

► **Converted in place**

This conversion is achieved by using the DFSMSdss CONVERTV function. This approach requires exclusive use of the data sets residing on the disk device. If data sets are already positioned in pools of volumes, this method might be an appropriate to use (table spaces are likely to be grouped this way). Be warned, if the volume and data sets do not meet all SMS requirements, DFSMSdss sets the volume's physical status to INITIAL. This status allows data sets to be accessed, but not extended. New allocations on the volume are prevented. If all requirements are met, DFSMSdss sets the volume status to CONVERTED.

***Tailor online conversion***

Many installations have data sets that are open and active most of the time. Staging a conversion into smaller manageable portions of data provides safer implementation results.

***Contingency time frame***

Limit the amount of data converted at a particular time, so if problems are experienced, the situation can be recovered or backed out.

**SMS implementation**

The storage administrator performs the implementation of SMS, using ISMF to update the ACS routines. However, the DB2 administrator is normally the person who is closely involved with the planning and positioning of data. This section outlines the required activities.

***Definition of Data Classes***

Although a good practice is for Data Classes to be assigned, it is optional. Even though it is not saved for non SMS-managed data sets, the allocation attributes in the Data Class are used to allocate the data set.

***Definition of Storage Classes***

Data sets must have a Storage Class to qualify for SMS management. Here, GUARANTEED SPACE is specified, along with availability, performance, and accessibility characteristics.

***Definition of Management Classes***

This definition is used for migration to level 1 and level 2 with or without backup, and indicates whether HSM management is used or not used (for backup or migration). It also includes expiration of data sets and space release/compaction.

***Definition of Storage Groups***

The storage group contains volumes that satisfy the service requirements of the data sets allocated to them. They can handle more than one type of data. Separate storage groups must be defined for production table spaces, active logs, other production data, and non-production data.

***Policy documentation***

The storage administrator defines the following policies:

- Data set naming conventions
- Volume naming conventions
- Restrictions on use of volumes
- The mapping of DB2 STOGROUPS with SMS storage groups



- ▶ Use of Data Classes
- ▶ Use of GUARANTEED SPACE parameter
- ▶ Use of Storage Classes
- ▶ Use of Management Classes

### **ACS routines**

The storage administrator uses the agreed policies to implement DB2 data under the control of SMS, which must be a documented procedure that includes the following items:

- ▶ Taking copies of the SMS control data set (ACDS, SCDS), and the source of the ACS routines prior to updating, for back out purposes
- ▶ Adding the relevant code for the DB2 data to the ACS routines
- ▶ Translating and validating the ACS routines
- ▶ Generating test cases, to ensure updates to the ACS routines have the desired effect
- ▶ Activating the new SMS configuration

### **Post implementation**

After DB2 data sets are SMS-managed, an ongoing procedure must be in place for maintaining the environment:

- ▶ Monitoring performance, availability, and accessibility
- ▶ Ensuring that DB2 data receives the correct level of service

The use of monitoring tools such as ISMF, CLISTS, and DFSMS Optimizer<sup>1</sup> can be used to help achieve these goals.

## **5.4.8 DFSMS FIT**

DFSMS Fast Implementation Techniques (FIT) is a process that supports implementation of SMS. The process was developed after working with a number of DFSMS implementations, and provides a proven design that can lead to a successful SMS implementation within several weeks.

Most installations implement SMS on a phased basis. First, candidates such as batch and TSO data can be targeted. After gaining operational experience, other categories such as databases can be included.

With DFSMS FIT, a complete design can be developed, and then the implementation can be phased in manageable portions. It uses a question and answer approach for steps of the design process. The documentation includes many samples of implementation, including jobs, coding, and procedures.

The process assumes IBM NaviQuest for MVS will be used for testing.

For more information about the implementation techniques, see the following publications:

- ▶ *Get DFSMS FIT: Fast Implementation Techniques*, SG24-2568
- ▶ *DFSMS FIT: Fast Implementation Techniques Process Guide*, SG24-4478
- ▶ *DFSMS FIT: Fast Implementation Techniques Installation Examples*, SG24-2569

<sup>1</sup> See [http://www.ibm.com/systems/storage/software/opt/overview/input\\_data.html](http://www.ibm.com/systems/storage/software/opt/overview/input_data.html).

## 5.4.9 NaviQuest

IBM NaviQuest for MVS can be used in conjunction with DFSMS FIT. It is a testing and reporting tool for the DFSMS environment, and is designed specifically to work with DFSMS FIT. It provides the following facilities:

- ▶ Automatically test the DFSMS configuration.
- ▶ Automatically create test cases.
- ▶ Automatically test the ACS routines.
- ▶ Perform storage reporting, through ISMF and with DCOLLECT and Volume Mount Analyzer (VMA) data.
- ▶ Print ISMF lists.
- ▶ Run ISMF functions in batch mode, using the REXX EXECs provided.

For more information about this feature, see the following publications:

- ▶ Chapter 21 in *DFSMS/MVS V1R3 NaviQuest User's Guide*, SC26-7194
- ▶ "Using NaviQuest" in *z/OS V1R9.0 DFSMS Storage Administration Reference (for DFSMSdfp, DFSMSdss, DFSMSHsm)*, SC26-7402-08



## DB2 I/O performance and monitoring

The DB2 instrumentation facility component (IFC) provides a trace facility that you can use to record DB2 data and events. Each trace class captures information about several subsystem events. These events are identified by many instrumentation facility component identifiers (IFCIDs). The IFCIDs are described by the comments in their mapping macros, contained in prefix.SDSNMACS, which is shipped to you with DB2.

Analysis and reporting of the trace records must take place outside of DB2. You can use IBM Tivoli OMEGAMON XE for DB2 Performance Expert for z/OS to format, print, and interpret DB2 trace output. You can view an online snapshot from trace records by using OMEGAMON PE or other online monitors. You can also consider writing your own program using the instrumentation facility interface (IFI). This chapter addresses I/O performance reporting and monitoring tools in relation to storage management in a DB2 environment.

This chapter describes the following tools:

- ▶ Tivoli OMEGAMON DB2 Performance Expert (OMEGAMON PE)
- ▶ Resource Measurement Facility (RMF)

See the following resources:

- ▶ For information about the trace categories and how to set up and interpret DB2 trace output, see *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851.
- ▶ For planning and monitoring tools with the DS8000, see *DS8000 Performance Monitoring and Tuning*, SG24-7146-01.
- ▶ For OMEGAMON PE documentation, go to the following web address:  
[http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?topic=/com.ibm.omegamon.xe\\_db2.doc/ko2rcd2025.htm](http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?topic=/com.ibm.omegamon.xe_db2.doc/ko2rcd2025.htm)
- ▶ For the z/OS Resource Measurement Facility (RMF), go to the following web address:  
<http://www.ibm.com/systems/z/os/zos/features/rmf/>

## 6.1 Global view of a DB2 I/O

Each time an application reads or writes data, DB2 requires or updates pages in buffer pools. DB2, synchronously or asynchronously, issues I/O requests, which can trigger synchronous or asynchronous staging and destaging operations between cache and physical disks. Figure 6-1 displays the relationships between DB2 and the storage server views of an I/O request from or to disk.

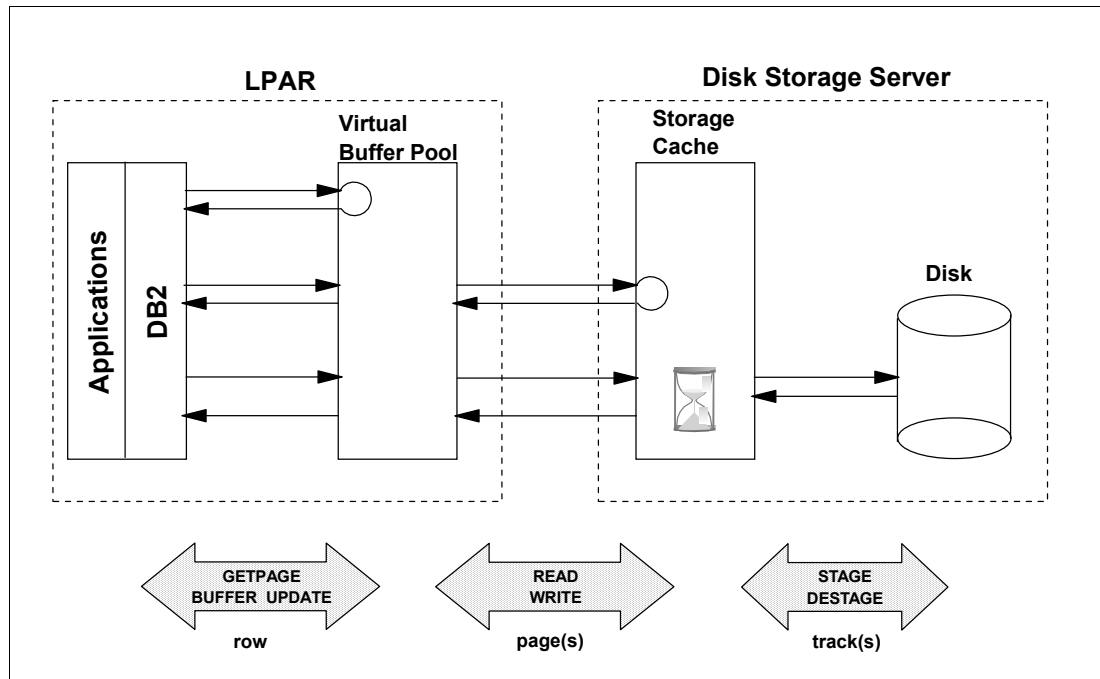


Figure 6-1 Scope of performance analysis tools

You can obtain OMEGAMON reports of accounting data in long or short format and in various levels of detail.

The examples of OMEGAMON reports in this information are based on the default formats, which might have been modified for your installation.

See the OMEGAMON Report Reference for an exact description of each report. You can also time the results for nested activities such as triggers, stored procedures, and user-defined functions.

The OMEGAMON accounting report, short format, allows you to monitor application distribution, resources used by each major group of applications, and the average DB2 elapsed time for each major group.

The report summarizes application-related performance data and orders the data by selected DB2 identifiers. Monitoring application distribution helps you to identify the most frequently used transactions or queries, and is intended to cover the 20% of the transactions or queries that represent about 80% of the total work load. Use the TOP list function of OMEGAMON to identify the report entries that represent the largest user of a given resource.

To get an overall picture of the system workload, you can use the OMEGAMON GROUP command to group several DB2 plans together.

The special synergy between disk subsystems and the System z operating systems (mainly the z/OS operating system) makes the disk storage servers outstanding performers in that environment.

The following list shows specific DS8000 performance features as they relate to application I/O in a z/OS environment that are beneficial to DB2:

- ▶ Parallel access volumes (PAVs)
- ▶ Multiple allegiance
- ▶ I/O priority queuing
- ▶ Logical volume sizes
- ▶ Fibre Channel connection (FICON)

These functions are described in Chapter 2, “Disk and tape environment overview” on page 11.

Several tools are available to help with monitoring the performance of the DS8000 features:

- ▶ Resource Measurement Facility (RMF)
- ▶ RMF Magic<sup>1</sup>
- ▶ The OMEGAMON family of IBM Tivoli products:

<http://www.ibm.com/software/tivoli/>

In this chapter, we provide examples of RMF usage and introduce RMF Magic.

---

<sup>1</sup> In July 2010 RMF Magic was enhanced and renamed to IntelliMagic Vision to reflect a more general use of the tool. See <http://www.intellimagic.net/intellimagic/products/rmf-magic>

## 6.2 Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

DB2 generates data about its own performance, called *instrumentation data*, but it has no reporting facility to analyze this data. The entries in the DB2 installation panel DSNTIPN, reported as shown in Figure 6-2, activate audit, global, accounting, and monitoring traces.

DSNTIPN		MIGRATE DB2 - TRACING PARAMETERS	
====>			
Enter data below:			
1	AUDIT TRACE	==> NO	Audit classes to start. NO,YES,list
2	TRACE AUTO START	==> NO	Global classes to start. YES, NO, list
3	TRACE SIZE	==> 65536	Trace table size in bytes. 4K-396K
4	SMF ACCOUNTING	==> 1	Accounting classes to start. NO,YES,list
5	SMF STATISTICS	==> YES	Statistics classes to start. NO,YES,list
6	STATISTICS TIME	==> 15	Time interval in minutes. 1-1440
7	STATISTICS SYNC	==> 0	Synchronization within the hour. NO,0-59
8	DATASET STATS TIME	==> 5	Time interval in minutes. 1-1440
9	MONITOR TRACE	==> NO	Monitor classes to start. NO, YES, list
10	MONITOR SIZE	==> 256K	Default monitor buffer size. 256K-16M
11	UNICODE IFCIDS	==> NO	Include UNICODE data when writing IFCIDS
12	DDF/RRSAF ACCUM	==> NO	Rollup accting for DDF/RRSAF. NO,2-65535
13	AGGREGATION FIELDS	==> 0	Rollup accting aggregation fields, 0-17
PRESS: ENTER to continue RETURN to exit HELP for more information			

Figure 6-2 Installation panel DSNTIPN

We have already seen in previous chapters that IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS (OMEGAMON PE for short in this publication) provides the capability to gather, analyze, and report on DB2 instrumentation data. OMEGAMON PE can report performance information online and in batch.

See the following publications:

- ▶ For information about the DB2 trace categories, see *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851.
- ▶ For the reporting functions, see *IBM Tivoli OMEGAMON XE for DB2 Performance Expert and Performance Monitor on z/OS Configuration and Customization*, GC19-2511.

The DB2 instrumentation data creates several types of trace records. The traces relevant to I/O analysis are as follows:

- ▶ Accounting trace
- ▶ Statistics trace
- ▶ Performance trace

Accounting and statistics traces are collected in most installations. A performance trace is collected when a specific problem has to be investigated. Activating the performance trace has a significant affect on DB2 subsystem performance. The user can cause more or less information to be collected by these traces, by specifying trace classes to be activated.

An *accounting trace* provides information at an identifier level. Examples of identifiers are plans, packages, users, or connection types. Accounting information can be a summary of multiple executions (Accounting Report), or it can be a detail of every execution (Accounting Trace).

A *statistics trace* provides information at DB2 subsystem level. It can be a summary of multiple statistic intervals (Statistics Report) or it can be a listing of each interval (Statistics Trace).

The *performance trace* can generate detailed information about all DB2 subsystem activity. When this trace has been started with the appropriate classes<sup>2</sup>, OMEGAMON PE can generate an I/O activity report.

## 6.2.1 Accounting I/O information

Accounting information includes I/O operations and I/O suspensions.

### I/O operations

The data I/O operations initiated by DB2 on behalf of an application are detailed in the buffer pool sections of the OMEGAMON PE accounting report. Each buffer pool is shown independently. For an example, see Figure 6-3.

Every read-access is reported as a getpage. Every write is reported as a buffer update. The getpages and buffer updates may initiate an I/O. These I/Os are either synchronous or asynchronous. The asynchronous reads are reported in the three prefetch fields (SEQ, LIST, DYN).

BP4	TOTAL
-----	-----
BPOOL HIT RATIO (%)	50
GETPAGES	300350
BUFFER UPDATES	0
SYNCHRONOUS WRITE	0
SYNCHRONOUS READ	754
SEQ. PREFETCH REQS	702
LIST PREFETCH REQS	0
DYN. PREFETCH REQS	3944
PAGES READ ASYNCHR.	148634
HPOOL WRITES	0
HPOOL WRITES-FAILED	0
PAGES READ ASYN-HPOOL	0
HPOOL READS	0
HPOOL READS-FAILED	0

Figure 6-3 OMEGAMON PE accounting, buffer pool section

### I/O suspensions

If accounting trace class 3 is activated, the DB2 accounting reports show a summary of the wait-times of a DB2 application. This summary shows when the DB2 application is suspended (waiting) for a DB2 system task to complete. These suspensions include the waits for I/O operations.

<sup>2</sup> IFCID 199 for buffer pool statistics at data set level, performance trace class 4 for database I/O, and class 5 for logging I/O.

Figure 6-4 shows an example of class 3 suspend-times. The values shown for A, B, C are total elapsed times for I/O and the number of occurrences of each type of I/O (events):

- A:** SYNCHRON. I/O      The total elapsed time as a result of synchronous I/O and the total number of synchronous I/O suspensions.
- B:** OTHER READ I/O      The total waiting time as a result of asynchronous read I/O and the total number of suspensions as a result of asynchronous read I/O.
- C:** OTHER WRTE I/O      The total elapsed time as a result of asynchronous write I/O and the total number of asynchronous write I/O suspensions.

CLASS 3 SUSP.	ELAPSED TIME	EVENTS	
-----	-----	-----	
LOCK/LATCH	0.060799	2760	
SYNCHRON. I/O	56.770913	19559	<b>A</b>
OTHER READ I/O	29:59.155952	143669	<b>B</b>
OTHER WRTE I/O	0.000000	0	<b>C</b>
SER.TASK SWTCH	0.002627	2	
ARC.LOG(QUIES)	0.000000	0	
ARC.LOG READ	0.000000	0	
DRAIN LOCK	0.000000	0	
CLAIM RELEASE	0.000000	0	
PAGE LATCH	0.000000	0	
STORED PROC.	0.000000	0	
NOTIFY MSGS	0.000000	0	
GLOBAL CONT.	0.000000	0	
TOTAL CLASS 3	30:55.990291	165990	

Figure 6-4 OMEGAMON PE accounting class 3 times

Another example of *other write I/O* is the case in which a transaction is waiting because the page that the transaction wants to update is currently being written out by a write engine. In this case, the wait time is captured in other write I/O.

Wait time for force at commit time is included in SYNC I/O WAIT and LOG FORCE WRITE WAIT.

## 6.2.2 Statistics I/O information

Statistics information includes data I/O operations and log activity.

### Data I/O operations

The statistics report provides detailed information about I/O operations for each buffer pool at the DB2 subsystem level. This means that this report summarizes the information of all applications executing during the interval that statistics are gathered. The information provided is much more detailed than the information in the accounting report.

Other additional information is calculated in this report, for example, Figure 6-5 on page 323 shows the average number of pages for each type of prefetch read.



BP4	READ OPERATIONS	QUANTITY	/MINUTE	/THREAD	/COMMIT
-----	-----	-----	-----	-----	-----
BPOOL	HIT RATIO (%)	55.12			
	GETPAGE REQUEST	221.8K	6534.43	N/C	110.9K
	GETPAGE REQUEST-SEQUENTIAL	18427.00	542.99	N/C	9213.50
	GETPAGE REQUEST-RANDOM	203.3K	5991.43	N/C	101.7K
	SYNCHRONOUS READS	613.00	18.06	N/C	306.50
	SYNCHRON. READS-SEQUENTIAL	64.00	1.89	N/C	32.00
	SYNCHRON. READS-RANDOM	549.00	16.18	N/C	274.50
	GETPAGE PER SYN.READ-RANDOM	370.36			
	SEQUENTIAL PREFETCH REQUEST	577.00	17.00	N/C	288.50
	SEQUENTIAL PREFETCH READS	577.00	17.00	N/C	288.50
	PAGES READ VIA SEQ.PREFETCH	18440.00	543.38	N/C	9220.00
	S.PRF.PAGES READ/S.PRF.READ	31.96			
	LIST PREFETCH REQUESTS	0.00	0.00	N/C	0.00
	LIST PREFETCH READS	0.00	0.00	N/C	0.00
	PAGES READ VIA LIST PREFETCH	0.00	0.00	N/C	0.00
	L.PRF.PAGES READ/L.PRF.READ	N/C			
	DYNAMIC PREFETCH REQUESTED	2515.00	74.11	N/C	1257.50
	DYNAMIC PREFETCH READS	2515.00	74.11	N/C	1257.50
	PAGES READ VIA DYN.PREFETCH	80470.00	2371.23	N/C	40.2K
	D.PRF.PAGES READ/D.PRF.READ	32.00			
	PREF.DISABLED-NO BUFFER	0.00	0.00	N/C	0.00
	PREF.DISABLED-NO READ ENG	0.00	0.00	N/C	0.00
	SYNC.HPOOL READ	0.00	0.00	N/C	0.00
	ASYNCH.HPOOL READ	0.00	0.00	N/C	0.00
	HPOOL READ FAILED	0.00	0.00	N/C	0.00
	ASYN.DA.MOVER HPOOL READ-S	0.00	0.00	N/C	0.00
	ASYN.DA.MOVER HPOOL READ-F	0.00	0.00	N/C	0.00
	PAGE-INS REQUIRED FOR READ	59.00	1.74	N/C	29.50

Figure 6-5 OMEGAMON PE statistics, buffer pool read operations section

## Log activity

The log activity report gives detailed information about log I/O. Figure 6-6 on page 324 shows a log activity section from a statistics report:

- The block of lines identified by **A** indicates read access. Log-reads are necessary for rollback, restart, and database recovery because DB2 must read from the log and apply changes to the data on disk. The various values indicate statistics for each possible source of log records. For performance reasons, most of the read accesses are satisfied from the output buffer or the active log; the archive log must be used only for exceptional circumstances.
- The block of lines identified by **B** indicates write access. Log records are created and written to the log when application programs update data. Each data update requires two

log records, one with the data before the update, and another with the data after the update, generally combined into one physical log write record.

- Line **C** shows BSDS accesses. As with the active log accesses, these accesses are mainly writes.
- The block of lines starting with **D** shows volume of records created in the active log and offloaded by the archiving process.
- The block of lines starting with **E** shows archive volume mounting information if tapes are used for archiving.

LOG ACTIVITY		QUANTITY	/MINUTE	/THREAD	/COMMIT
-----		-----	-----	-----	-----
READS SATISFIED-OUTPUT BUFF	<b>A</b>	0.00	0.00	N/C	0.00
READS SATISFIED-OUTP.BUF(%)		N/C			
READS SATISFIED-ACTIVE LOG		0.00	0.00	N/C	0.00
READS SATISFIED-ACTV.LOG(%)		N/C			
READS SATISFIED-ARCHIVE LOG		0.00	0.00	N/C	0.00
READS SATISFIED-ARCH.LOG(%)		N/C			
TAPE VOLUME CONTENTION WAIT		0.00	0.00	N/C	0.00
WRITE-NOWAIT	<b>B</b>	0.00	0.00	N/C	0.00
WRITE OUTPUT LOG BUFFERS		509.00	15.00	N/C	254.50
BSDS ACCESS REQUESTS	<b>C</b>	0.00	0.00	N/C	0.00
UNAVAILABLE OUTPUT LOG BUFF		0.00	0.00	N/C	0.00
CONTR.INTERV.CREATED-ACTIVE	<b>D</b>	5.00	0.15	N/C	2.50
ARCHIVE LOG READ ALLOCATION		0.00	0.00	N/C	0.00
ARCHIVE LOG WRITE ALLOCAT.		0.00	0.00	N/C	0.00
CONTR.INTERV.OFFLOADED-ARCH		0.00	0.00	N/C	0.00
READ DELAYED-UNAVAIL.RESOUR	<b>E</b>	0.00	0.00	N/C	0.00
LOOK-AHEAD MOUNT ATTEMPTED		0.00	0.00	N/C	0.00
LOOK-AHEAD MOUNT SUCCESSFUL		0.00	0.00	N/C	0.00

Figure 6-6 OMEGAMON PE statistics, log activity section

### 6.2.3 Performance I/O information and I/O activity

OMEGAMON PE can create several I/O activity reports for various types of data sets. The buffer pool report shows the activity of one identifier (for example: plan, user, buffer pool identifier) against the data sets in one virtual buffer pool.

Before the report can be generated, the appropriate trace must be started. Limit the trace to reduce the impact on system performance by specifying class and instrumentation facility component identifiers (IFCIDs) in the START TRACE command. Asynchronous I/O activity is not collected when user (AUTHID) or application identifiers (PLAN) are specified in the START TRACE command unless the user identifier of the DBM1 address space is also listed.

Table 6-1 shows the requirements to generate the trace information for the different I/O activity reports.

*Table 6-1 Trace requirement for the I/O activity reports*

I/O activity report	DB2 trace	Class	IFCID
Buffer pool	Performance	4	6, 7, 8, 9, 10, 105, 107
EDM pool	Performance	4	29, 30, 105, 107
Active log	Performance	5	34, 35, 36, 37, 38, 39
Archive log and BSDS	Performance	5	34, 35, 36, 37, 40, 41, 114, 115, 116, 119, 120
Cross invalidation	Performance	21	105, 107, 255

Figure 6-7 shows an extract from a summary I/O activity report. A detail report allows an analysis at the identifier level. For example, it can be used for a detailed analysis of the accesses of one application to one table space.

BUFFER POOL	TOTALS	AET
-----	-----	-----
TOTAL I/O REQUESTS	51	0.019885
TOTAL READ I/O REQUESTS	51	0.019885
NON-PREFETCH READS	51	
PREFETCH READS		
WITHOUT I/O	0	
WITH I/O	0	
PAGES READ	0	
PAGES READ / SUCC READ	0.00	
TOTAL WRITE REQUESTS	0	
SYNCHRONOUS WRITES	0	N/C
COUPLING FACILITY CASTOUTS	0	N/C
PAGES WRITTEN PER WRITE	0.00	
ASYNCHRONOUS WRITES	0	N/C
COUPLING FACILITY CASTOUTS	0	N/C
PAGES WRITTEN PER WRITE	0.00	

*Figure 6-7 Buffer pool section from I/O activity summary report*

For specific issues, you might need to activate a specific performance trace. Trace descriptions and trace types for your DB2 version are listed in the following library member:

h1q.SDSNIVPD(DSNWMSGs)

## 6.3 RMF monitoring

Resource Measurement Facility (RMF), which is part of the z/OS operating system, provides performance information about the IBM disk subsystems to the users. For the current IBM System Storage DS8000 series (M/T 2107), APAR OA06476 provides full support RMF reporting.

Three types of time intervals are used by RMF monitors for data gathering on z/OS systems:

- ▶ Long term data gathering with Monitor I and Monitor III
- ▶ Snapshot™ monitoring with Monitor II
- ▶ Short term data collection with Monitor III

All three monitors write SMF records (type 70 - type 79) if you define the appropriate SMF recording options.

RMF can help with monitoring the following performance components:

- ▶ I/O response time
- ▶ PAV
- ▶ IOP/SAP
- ▶ FICON host channel
- ▶ FICON director
- ▶ Cache and NVS
- ▶ FICON/Fibre port and host adapter
- ▶ Extent pool and rank

For more information, see *z/OS Resource Measurement Facility Report Analysis*, SC33-7991.

We examine these components in the following sections.

### 6.3.1 I/O response time

The RMF DIRECT ACCESS DEVICE ACTIVITY report (Example 6-1) is usually the first report to use to monitor performance of the disk subsystems by providing details about I/O response time<sup>3</sup>. If a service level agreement (SLA) is not being met and the problem might be related to storage, this report is the starting point in performance analysis. If possible, rank volumes related to the application by I/O intensity, which is the I/O rate multiplied by Service Time (the Service Time = PEND + DISC + CONN time).

The device activity report (created by RMF Monitor I with SMF record type 74 subtype 1) accounts for all activity to a base and all of its associated alias addresses. See Example 6-1.

*Example 6-1 RMF direct access device activity report*

D I R E C T   A C C E S S   D E V I C E   A C T I V I T Y																						
STORAGE GROUP	DEV NUM	DEVICE TYPE	NUMBER OF CYL	VOLUME SERIAL	PAV	LCU	DEVICE	AVG	AVG	AVG	AVG	AVG	AVG	AVG	%	%	%	AVG	%	%		
							ACTIVITY RATE	RESP TIME	IOSQ TIME	CMR DLY	DB DLY	PEND TIME	DISC TIME	CONN TIME	DEV CONN	DEV UTIL	DEV RESV	NUMBER ALLOC	ANY ALLOC	MT PEND		
DBS0	A100	33903	3339	AISL00	4	007E	0.017	1.39	.000	.128	.000	.213	.836	.341	0.00	0.00	0.0	0.0	100.0	0.0		
	A101	33903	3339	AISL01	4	007E	0.014	1.29	.000	.128	.000	.207	.788	.295	0.00	0.00	0.0	0.0	100.0	0.0		
	A102	33909	30051	D26454	5	007E	0.198	5.39	.000	.152	.000	.252	3.21	1.92	0.01	0.02	0.0	10.0	100.0	0.0		
	A103	33909	30051	D26455	5	007E	11.218	1.47	.000	.140	.000	.237	.846	.389	0.09	0.28	0.0	9.0	100.0	0.0		
	A104	33909	30051	D26456	5	007E	0.990	2.73	.000	.144	.000	.242	1.89	.597	0.01	0.05	0.0	6.0	100.0	0.0		
DBS0	A105	33909	30051	D26457	5	007E	0.903	12.9	.000	.140	.000	.237	12.0	.640	0.01	0.23	0.0	11.0	100.0	0.0		
DBS1	A106	33909	30051	D26458	8*	007E	2.654	.803	.000	.139	.000	.238	.228	.337	0.01	0.02	0.0	18.0	100.0	0.0		
PRD0	A107	33909	10017	P26531	5	007E	0.154	2.53	.000	.149	.000	.250	.492	1.78	0.01	0.01	0.1	0.0	100.0	0.0		
PRD0	A108	33909	10017	P26532	5	007E	0.741	6.50	.000	.142	.000	.245	4.96	1.30	0.02	0.09	0.0	10.0	100.0	0.0		
PRD0	A109	33909	10017	P26533	5	007E	0.186	3.31	.000	.136	.000	.232	2.58	.497	0.00	0.01	0.0	7.0	100.0	0.0		
PRD0	A10A	33909	10017	P26534	5	007E	0.111	1.60	.000	.133	.000	.224	.914	.463	0.00	0.00	0.0	1.0	100.0	0.0		

Activity about alias addresses is not reported separately, but the alias addresses are accumulated into the base address.

<sup>3</sup> For details about the components, see "What are the components of I/O response time?" on page 376.

Starting with z/OS Release 1.10, this report also shows the number of cylinders allocated to the volume.

## 6.3.2 PAV

PAV is the number of addresses assigned to a UCB, which includes the base address plus the number of aliases assigned to that base address.

RMF reports the number of PAV addresses (or in RMF terms, *exposures*) that have been used by a device. In a dynamic PAV environment, when the number of exposures has changed during the reporting interval, there is an asterisk next to the PAV number. Example 6-1 on page 326 shows that address A106 has a PAV of 8\*, the asterisk indicates that the number of PAVs was either lower or higher than 8 during the previous RMF period.

For HyperPAV, the number of PAVs is shown in this format: *n.nH*. The *H* indicates that this volume is supported by HyperPAV, and the *n.n* is a one decimal number showing the average number of PAVs assigned to the address during the RMF report period. When a volume has no I/O activity, the PAV is always 1, which means that there is no alias assigned to this base address, because in HyperPAV, an alias is used or assigned to a base address only during the period required to execute an I/O. The alias is then released and put back into the alias pool after the I/O is completed.

**Note:** The number of PAVs includes the base address plus the number of aliases assigned to it. Thus, a PAV=1 means that the base address has no aliases assigned to it.

## 6.3.3 IOP/SAP

The IOP/SAP<sup>4</sup> is the central electronics complex (CEC) processor that handles the I/O operation. Check the I/O queuing activity report (Figure 6-8) to determine whether the IOP is saturated.

-	LCU	CU	DCM	GROUP	CHAN	CHPID	% DP	% CU	AVG	AVG	CONTENTION	DELAY	AVG	HPAV	
			MIN	MAX	DEF	PATHS	TAKEN	BUSY	CUB	CMR	RATE	Q	CSS	WAIT	MAX
0 0003	0070					68	0.482	0.00	0.00	0.0	0.0				
						*	0.482	0.00	0.00	0.0	0.0				
0 000D	1000					90	5.015	0.00	0.00	0.0	0.0				
						91	5.101	0.00	0.00	0.0	0.0				
						92	5.162	0.00	0.00	0.0	0.0				
						94	*								
						*	15.278	0.00	0.00	0.0	0.0	OFFLINE			
							9.423	0.00	0.00	0.0	0.0	0.000	0.00	0.2	0
0 000E	1100					10	8.755	0.00	0.00	0.0	0.0				
						11	9.399	0.00	0.00	0.0	0.0				
			1	1	3	*	27.578	0.00	0.00	0.0	0.0				
						*	3.667	0.00	0.00	0.0	0.0	0.000	0.00	0.4	
0 000F	1200					90	3.526	0.00	0.00	0.0	0.0				
						91	3.659	0.00	0.00	0.0	0.0				
						92									
						94									
						*	10.852	0.00	0.00	0.0	0.0	OFFLINE			
												0.000	0.00	0.2	0

Figure 6-8 I/O queuing activity report

This report is created by RMF Monitor I with SMF record type 78 subtype 3. An *average queue length* greater than 1 indicates that the IOP is saturated, even though an average queue length greater than 0.5 is considered as a warning sign. A burst of I/O can also trigger a high average queue length.

<sup>4</sup> A system assist processor (SAP) is a processing unit that runs the channel subsystem Licensed Internal Code to control input/output (I/O) operations.

If only certain IOPs are saturated, redistributing the channels assigned to the disk subsystems can help balance the load to the IOP, because an IOP is assigned to handle a certain set of channel paths. Therefore, assigning all the channels from one IOP to access a busy disk subsystem can cause a saturation on that particular IOP.

### 6.3.4 FICON host channel

The FICON report in Example 6-2 shows the FICON channel-related statistics.

*Example 6-2 Channel Path Activity report*

CHANNEL PATH ACTIVITY																									
z/OS V1R6				SYSTEM ID SYS1				DATE 06/15/2005				INTERVAL 00.59.997													
				RPT VERSION V1R5 RMF				TIME 11.34.00				CYCLE 1.000 SECONDS													
IODF = 94		CR-DATE: 06/15/2005		CR-TIME: 09.34.25		ACT: ACTIVATE		MODE: LPAR		CPMF: EXTENDED MODE															
-----																									
DETAILS FOR ALL CHANNELS																									
-----																									
CHANNEL		PATH		UTILIZATION(%)			READ(MB/SEC)			WRITE(MB/SEC)			CHANNEL		PATH		UTILIZATION(%)			READ(MB/SEC)			WRITE(MB/SEC)		
ID	TYPE	G	SHR	PART	TOTAL	BUS	PART	TOTAL	PART	TOTAL	ID	TYPE	G	SHR	PART	TOTAL	BUS	PART	TOTAL	PART	TOTAL	PART	TOTAL		
2E	FC_S	2	Y	0.15	0.66	4.14	0.02	0.13	0.05	0.08	36	FC_?													
2F	FC_S	2	Y	0.15	0.66	4.14	0.02	0.12	0.05	0.08	37	FC_?													
30	FC_S	2	Y	0.02	0.14	3.96	0.00	0.00	0.00	0.00	38	FC_S	2	Y	0.00	5.17	4.45	0.00	0.00	0.00	0.00	0.00			
31	FC_S	2	Y	0.02	0.14	3.96	0.00	0.00	0.00	0.00	39	FC_S	2	Y	0.00	4.47	4.37	0.00	0.00	0.00	0.00	0.00			
32	FC_?	2	Y	0.00	0.13	3.96	0.00	0.00	0.00	0.00	3A	FC_S	2	Y	0.02	0.14	3.96	0.00	0.00	0.00	0.00	0.00			
3B	FC_S	2	Y	9.20	9.20	10.20	0.00	0.00	13.28	13.28	43	FC_S													
3C	FC_S	2	Y	3.09	3.14	6.53	6.37	6.37	0.00	0.00	44	FC_S	2	Y	9.27	9.27	10.37	0.00	0.00	14.07	14.07				
3D	FC_S	2	Y	3.25	3.31	6.50	6.34	6.34	0.00	0.00	45	FC	2		0.00	0.13	3.96	0.00	0.00	0.00	0.00				

The *PART Utilization* is the FICON channel utilization related to the I/O activity on this LPAR, and the *Total Utilization* is the total utilization of the FICON channel from all LPARs that are defined on the CEC. The general rule for the total FICON channel utilization and the BUS utilization is to keep them under 50%. If these numbers exceed 50%, you see an elongated connection (CONN) time.

For small block transfers, the BUS utilization is less than the FICON channel utilization, and for large block transfers, the BUS utilization is greater than the FICON channel utilization.

The Generation (G) field in the channel report tells you what combination of generation FICON channel is being used and the speed of the FICON channel link for this CHPID at the time of the machine initial program load (IPL). The G field does not include any information about the link between the director and the DS8000:

- ▶ G=5 means the link between the channel and the director runs at 4 Gbps, which is applicable for a FICON Express4 channel.
- ▶ G=4 means the link between the channel and the director runs at 2 Gbps, which is applicable for a FICON Express4 or FICON Express2 channel.
- ▶ G=3 means the link between the channel and the director runs at 1 Gbps, which is applicable to a FICON Express4 or FICON Express2 channel.
- ▶ G=2 means the link between the channel and the director runs at 2 Gbps, which is applicable to a FICON Express channel.
- ▶ G=1 means the link between the channel and the director runs at 1 Gbps, which is applicable to a FICON Express channel.

The link between the director and the DS8000 can run at 1, 2, or 4 Gbps.

If the channel is connected point-to-point to the DS8000 FICON port, the G field indicates the speed that was negotiated between the FICON channel and the DS8000 port.

### 6.3.5 FICON director

*FICON director* is the switch that is used to connect the host FICON channel to the DS8000 FICON port. FICON director performance statistics are collected in the System Management Facilities (SMF) record type 74 subtype 7. The FICON DIRECTOR ACTIVITY report (Example 6-3) provides information about director and port activities. This report assists in analyzing performance problems and in capacity planning.

*Example 6-3 FICON Director Activity report*

F I C O N   D I R E C T O R   A C T I V I T Y									
SWITCH DEVICE:0414		SWITCH ID:01		TYPE:005000		MODEL:001		MAN:MCD	
						PLANT:01		SERIAL:00000MK00109	
PORT ADDR	-CONNECTION- UNIT	ID	AVG FRAME PACING	AVG FRAME READ	SIZE WRITE	PORT BANDWIDTH (MB/SEC)		ERROR COUNT	
						--READ	----WRITE --		
05	CHP	FA	0	808	285	50.04	10.50	0	
07	CHP	4A	0	149	964	20.55	5.01	0	
09	CHP	FC	0	558	1424	50.07	10.53	0	
0B	CHP-H	F4	0	872	896	50.00	10.56	0	
12	CHP	D5	0	73	574	20.51	5.07	0	
13	CHP	C8	0	868	1134	70.52	2.08	1	
14	SWITCH	----	0	962	287	50.03	10.59	0	
15	CU	C800	0	1188	731	20.54	5.00	0	

The measurements that are provided for a port in this report include the I/O for the system on which the report is taken *and also include* all I/Os that are directed through this port, regardless of which LPAR requests the I/O.

The CONNECTION information in this report shows where this port is connected:

- ▶ CHP: The port is connected to a FICON channel on the host.
- ▶ CHP-H: The port is connected to a FICON channel on the host that requested this report.
- ▶ CU: This port is connected to a port on a disk subsystem.
- ▶ SWITCH: This port is connected to another FICON director.

The important performance metric here is the AVG FRAME PACING. This metric shows the average time (in microseconds) that a frame had to wait before it was transmitted. The higher the contention on the director port, the higher the average frame pacing will be.

### 6.3.6 Cache and NVS

The RMF CACHE reports, based on SMF record type 74 subtype 5, provide useful information for analyzing the reason of high DISC time at a SUMMARY, SUBSYS, and DEVICE level. Figure 6-9 shows a sample cache subsystem summary report by LCU.

C A C H E   S U B S Y S T E M   S U M M A R Y																			PAGE		1
z/OS V1R10					SYSTEM ID   SYS1					DATE 07/28/2009					INTERVAL 15.00.055						
					RPT VERSION   VIR10 RMF					TIME 00.30.00											
SSID	CU-ID	TYPE	CACHE	NVS	I/O RATE	OFF RATE	--CACHE READ	HIT DFW	RATE- CFW	-----DASD STAGE	I/O DFWBP	RATE- ICL	----- BYP	OTHER	ASYNC RATE	TOTAL H/R	READ H/R	WRITE H/R	% READ		
3000	1B30	2105-E20	3072	192	67.5	0.0	23.8	38.8	0.0	4.9	0.0	0.0	0.0	0.0	20.4	0.928	0.830	1.000	42.5		
3001	1B87	2105-E20	3072	192	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A	N/A		
3002	1C1E	2105-E20	3072	192	27.2	0.0	23.3	0.0	0.0	3.9	0.0	0.0	0.0	0.0	0.0	0.857	0.857	N/A	100.0		
3003	1C95	2105-E20	3072	192	7.6	0.0	6.8	0.0	0.0	0.8	0.0	0.0	0.0	0.0	0.0	0.891	0.891	N/A	100.0		
3004	1D0C	2105-E20	3072	192	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A	N/A		

*Figure 6-9 Cache subsystem summary report*

Figure 6-10 shows the CACHE SUBSYSTEM OVERVIEW report by volume.

The report shows the I/O requests by read and by write. It shows the rate, the hit rate, and the hit ratio of the read and the write activities. The read-to-write ratio is also calculated. Note that the total I/O requests here can be higher than the I/O rate shown in the DASD report. In the DASD report, one channel program is counted as one I/O. However, in the cache report, if there is more than one **Locate Record** command in a channel program, each **Locate Record** command is counted as one I/O request.

CACHE SUBSYSTEM OVERVIEW															
OTOTAL I/O		29627	CACHE I/O		29627	CACHE OFFLINE		0							
TOTAL H/R		0.813	CACHE H/R		0.813										
---CACHE I/O REQUESTS---															
REQUESTS	COUNT	RATE	HITS	RATE	H/R	COUNT	RATE	FAST	RATE	HITS	RATE	H/R	READ	%	
ONORMAL	22168	375.7	16641	282.1	0.751	7459	126.4	7459	126.4	7459	126.4	1.000	74.8		
SEQUENTIAL	0	0.0	0	0.0	N/A	0	0.0	0	0.0	0	0.0	N/A	N/A		
CFW DATA	0	0.0	0	0.0	N/A	0	0.0	0	0.0	0	0.0	N/A	N/A		
OTOTAL	22168	375.7	16641	282.1	0.751	7459	126.4	7459	126.4	7459	126.4	1.000	74.8		
---CACHE MISSES---															
REQUESTS	READ	RATE	WRITE	RATE	TRACKS	RATE									
NORMAL	5527	93.7	0	0.0	5527	93.7									
SEQUENTIAL	0	0.0	0	0.0	0	0.0									
CFW DATA	0	0.0	0	0.0	0	0.0									
OTOTAL	5527	93.7	0	0.0	0	0.0									
---CKD STATISTICS---															
WRITE	0	READ MISSES		0											
WRITE HITS	0	WRITE FROM		5082											
1															
C A C H E S U B S Y S T E M A C T I V I T Y															
z/OS V1R6															
SYSTEM ID GDP2 START 06/07/2005-21.03.00 INTERVAL 000.00.59															
RPT VERSION V1R5 RMF END 06/07/2005-21.04.00															
OSUBSYSTEM 2107-01 CU-ID 8F01 CDATE 06/07/2005 CTIME 21.03.02 CINT 00.00.59															
TYPE-MODEL 2107-922 MANUF IBM PLANT 75 SERIAL 000000020331															
0															
CACHE SUBSYSTEM DEVICE OVERVIEW															
OVOLUME	DEV	XTNT	%	I/O	---CACHE HIT RATE---		---DASD I/O RATE---		ASYNC		TOTAL	READ	WRITE	%	
SERIAL	NUM	POOL	I/O	RATE	READ	DFW	CFW	STAGE	DFWBP	ICL	BYP	OTHER	H/R	H/R	READ
0*ALL			100.0	502.2	282.1	126.4	0.0	93.7	0.0	0.0	0.0	0.0	98.8	0.813	0.751
*CACHE-OFF			0.0	0.0											
*CACHE			100.0	502.2	282.1	126.4	0.0	93.7	0.0	0.0	0.0	0.0	98.8	0.813	0.751
PR8F00	8F00	001F	22.3	112.2	65.1	28.0	0.0	19.1	0.0	0.0	0.0	0.0	19.1	0.830	0.773
PR8F01	8F01	001F	11.3	56.6	31.9	14.2	0.0	10.5	0.0	0.0	0.0	0.0	10.6	0.815	0.753
PR8F02	8F02	001F	11.0	55.2	30.3	14.3	0.0	10.5	0.0	0.0	0.0	0.0	10.4	0.810	0.743
PR8F03	8F03	001F	11.2	56.2	31.8	14.0	0.0	10.4	0.0	0.0	0.0	0.0	10.6	0.815	0.754
PR8F04	8F04	001F	3.6	18.2	10.5	4.6	0.0	3.1	0.0	0.0	0.0	0.0	3.9	0.828	0.770
PR8F05	8F05	001F	3.5	17.8	9.5	4.5	0.0	3.8	0.0	0.0	0.0	0.0	3.9	0.789	0.717
PR8F06	8F06	001F	3.6	18.1	9.8	4.5	0.0	3.7	0.0	0.0	0.0	0.0	4.1	0.793	0.723
PR8F07	8F07	001F	3.9	19.8	11.1	5.0	0.0	3.7	0.0	0.0	0.0	0.0	4.3	0.812	0.749
PR8F08	8F08	001F	3.5	17.7	10.2	4.2	0.0	3.3	0.0	0.0	0.0	0.0	3.6	0.814	0.756
PR8F09	8F09	001F	3.7	18.8	10.4	4.7	0.0	3.6	0.0	0.0	0.0	0.0	4.1	0.806	0.741

Figure 6-10 Cache subsystem overview report

In this report, we can check to see the value of the read hit ratio. Low read-hit ratios contribute to higher DISC time. For a cache friendly workload, we see a read-hit ratio of better than 90%. The write-hit ratio is usually 100%.

High DFW BYPASS is an indication that persistent memory (NVS) is overcommitted. DFW BYPASS actually means DASD Fast Write I/Os that are retried, because persistent memory is full. Calculate the quotient of DFW BYPASS divided by the total I/O rate; as a general rule, if this number is higher than 1%, the write-retry operations have a significant impact on the DISC time.

Check the DISK ACTIVITY part of the report. The Read response time must be less than 35 ms. If it is higher than 35 ms, it is an indication that the DDMs on the rank where this LCU resides are saturated.

Figure 6-11 on page 331 shows the CACHE SUBSYSTEM DEVICE OVERVIEW report. Here, you can see to which extent pool each volume belongs. If we have the following setup, performing the analysis if a performance problem occurs on the LCU is easier:

- ▶ One extent pool has one rank.
- ▶ All volumes on an LCU belong to the same extent pool.



Looking at the rank statistics in the report in Example 6-6 on page 334, we know that all the I/O activity on that rank is from the same LCU. Therefore, we can concentrate our analysis on the volumes on that LCU only.

**Note:** Depending on the DDM size used and the 3390 model selected, you can put multiple LCUs on one rank, or you can also have an LCU that spans more than one rank.

C A C H E   S U B S Y S T E M   A C T I V I T Y															PAGE	2
z/OS V1R10				SYSTEM ID		SYS1	DATE 07/28/2009		INTERVAL 15.00.055							
				RPT VERSION		V1R10 RMF	TIME 00.30.00									
SUBSYSTEM	2105-01	CU-ID	1E91	SSID	3007	CDATE	07/28/2009	CTIME	00.30.00	CINT	15.00					
TYPE-MODEL	2105-E20	MANUF	IBM	PLANT	75	SERIAL	000000016374									
-----																
C A C H E   S U B S Y S T E M   D E V I C E   O V E R V I E W																
-----																
VOLUME	DEV	RRID	%	I/O	---CACHE	HIT	RATE---	-----DASD	I/O	RATE-----	ASYN	TOTAL	READ	WRITE	%	
SERIAL	NUM		I/O	RATE	READ	DFW	CFW	STAGE	DFWBP	ICL	BYP	OTHER	RATE	H/R	H/R	READ
*ALL			100.0	205.5	172.3	11.3	0.0	21.9	0.0	0.0	0.0	0.0	20.5	0.893	0.887	94.5
+CACHE-OFF			0.0	0.0												
+CACHE			100.0	205.5	172.3	11.3	0.0	21.9	0.0	0.0	0.0	0.0	20.5	0.893	0.887	94.5
NP1MHD	1E90	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
NP1MHE	1E81	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
NP1MHF	1E82	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
NP1MHG	1E83	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
NP1MHH	1E84	0700	8.1	16.5	14.7	0.0	0.0	1.8	0.0	0.0	0.0	0.0	0.0	0.891	0.891	100.0
NP1MHI	1E85	0700	17.3	35.6	27.9	0.0	0.0	7.6	0.0	0.0	0.0	0.0	0.0	0.785	0.785	100.0
NP1MHJ	1E86	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
NP1MHK	1E87	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
NP1MHL	1E88	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
NP1MHM	1E89	0700	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A
...																
-----																
R A I D   R A N K   A C T I V I T Y																
-----																
ID	RAID	DA	HDD	-----	READ REQ	-----	-----	WRITE REQ	-----	HIGHEST UTILIZED VOLUMES						
	TYPE			RATE	AVG MB	MB/S	RTIME	RATE	AVG MB	MB/S	RTIME					
*ALL			7	68	0.054	3.7	16	10	0.105	1.0	32					
0700	RAID-5	19	7	68	0.054	3.7	16	10	0.105	1.0	32	NP1MJ2	NP1MJK	NP1MHI	NP1MI4	NP1MHN

Figure 6-11 Cache Subsystem Activity by volume serial number

If you specify `REPORTS(CACHE(DEVICE))` when running the cache report, a detailed report by volume is generated, as in Example 6-4. This report gives you the detailed cache statistics of each volume. By specifying `REPORTS(CACHE(SSID(nnnn)))`, you can limit this report to only certain LCUs.

The report basically shows the same performance statistics as in Example 6-2 on page 328, but at the level of each volume.

**Example 6-4** Cache Device Activity report detail by volume

C A C H E   D E V I C E   A C T I V I T Y													
z/OS V1R9			SYSTEM ID WIN5			DATE 11/05/2008			INTERVAL 00.59.976				
			CONVERTED TO z/OS V1R10 RMF			TIME 21.54.00							
SUBSYSTEM	2107-01	CU-ID	C01C	SSID	0847	CDATE	11/05/2008	CTIME	21.54.01	CINT	01.00		
TYPE-MODEL	2107-932	MANUF	IBM	PLANT	75	SERIAL	0000000AB171						
VOLSER	@9C02F	NUM	C02F	extent	P00L	0000							
-----													
C A C H E   D E V I C E   S T A T U S													
-----													
CACHE STATUS						DUPLEX PAIR STATUS							
CACHING	- ACTIVE					DUPLEX PAIR		- NOT ESTABLISHED					
DASD FAST WRITE	- ACTIVE					STATUS		- N/A					
PINNED DATA	- NONE					DUAL COPY VOLUME		- N/A					
-----													
C A C H E   D E V I C E   A C T I V I T Y													
-----													
TOTAL I/O	3115	CACHE I/O	3115	CACHE OFFLINE	N/A								
TOTAL H/R	0.901	CACHE H/R	0.901										
CACHE I/O	-----												
REQUESTS	COUNT	RATE	HITS	RATE	H/R	COUNT	RATE	FAST	RATE	HITS	RATE	H/R	READ %
NORMAL	2786	46.4	2477	41.3	0.889	329	5.5	329	5.5	329	5.5	1.000	89.4
SEQUENTIAL	0	0.0	0	0.0	N/A	0	0.0	0	0.0	0	0.0	N/A	N/A
CFW DATA	0	0.0	0	0.0	N/A	0	0.0	0	0.0	0	0.0	N/A	N/A
TOTAL	2786	46.4	2477	41.3	0.889	329	5.5	329	5.5	329	5.5	1.000	89.4
-----													
REQUESTS	READ	RATE	WRITE	RATE	TRACKS	RATE	COUNT		RATE	-----NON-CACHE I/O-----			
										COUNT	RATE	COUNT	RATE
NORMAL	309	5.1	0	0.0	311	5.2	DFW BYPASS		0	0.0	ICL	0	0.0
SEQUENTIAL	0	0.0	0	0.0	0	0.0	CFW BYPASS		0	0.0	BYPASS	0	0.0
CFW DATA	0	0.0	0	0.0			DFW INHIBIT		0	0.0	TOTAL	0	0.0
TOTAL	309	RATE	5.1				ASYNC (TRKS)		173	2.9			
-----													
---CKD STATISTICS---		---RECORD CACHING---				---HOST ADAPTER ACTIVITY---				-----DISK ACTIVITY-----			
								BYTES /REQ /SEC		RESP TIME /REQ /SEC		BYTES /SEC	
WRITE	0	READ MISSES		0				4.1K 190.1K		READ 14.302		55.6K 288.4K	
WRITE HITS	0	WRITE PROM		111				4.0K 21.8K		WRITE 43.472		18.1K 48.1K	
-----													

## 6.3.7 FICON/Fibre port and host adapter

The report in Example 6-5 shows the port report on the DS8000, which includes the FICON ports and also the PPRC link ports. This report is created by RMF Monitor I with SMF record type 74 subtype 8.

**Example 6-5** DS8000 link statistics

E S S   L I N K   S T A T I S T I C S									
z/OS V1R7			SYSTEM ID SYSA			DATE 02/01/2008			INTERVAL 14.59.778
			CONVERTED TO z/OS V1R10 RMF			TIME 01.14.00			CYCLE 1.000 SECONDS
SERIAL NUMBER	00000ABC01	TYPE-MODEL	002107-921	CDATE	02/01/2008	CTIME	01.14.01	CINT	14.59
-----ADAPTER-----	--LINK TYPE--	BYTES /SEC	BYTES /OPERATION	OPERATIONS /SEC	RESP TIME /OPERATION	I/O INTENSITY			
0000 FIBRE 2Gb	ECKD READ	17.2M	9.9K	1735.2	0.1	131.6			
	ECKD WRITE	7.7M	14.5K	533.9	0.2	123.4			
						-----			
0001 FIBRE 2Gb	ECKD READ	9.1M	8.4K	1087.2	0.1	255.0			
	ECKD WRITE	7.7M	17.0K	455.9	0.2	79.9			
						-----			
0101 FIBRE 2Gb	PPRC SEND	6.0M	53.1K	112.2	9.1	181.2			
	PPRC RECEIVE	0.0	0.0	0.0	0.0	1024.9			
						0.0			

0102	FIBRE 2Gb							-----
		PPRC SEND	6.2M	53.1K	115.9	8.6	1024.9	
		PPRC RECEIVE	0.0	0.0	0.0	0.0	998.0	
							0.0	-----
0200	FIBRE 2Gb							998.0
		SCSI READ	10.8M	30.7K	352.4	0.2	67.5	
		SCSI WRITE	1.9M	31.5K	60.9	1.4	83.3	
								-----
0201	FIBRE 2Gb							150.8
		SCSI READ	9.0M	38.7K	232.0	0.2	53.3	
		SCSI WRITE	135.0K	10.7K	12.6	0.3	3.5	
								-----
								56.8

The SAID is the port ID:

- ▶ The first two characters denote the enclosure number (see Figure 6-12).
- ▶ The third character denotes the host adapter number within the enclosure:  
Numbered 0, 1, 3, and 4
- ▶ The last character denotes the port ID within that host adapter:  
Numbered 0, 1, 2, and 3

The report shows that the ports are running at 2 Gbps. There are FICON ports, shown under the heading of LINK TYPE as ECKD READ and ECKD WRITE, and there are also PPRC ports, shown as PPRC SEND and PPRC RECEIVE.

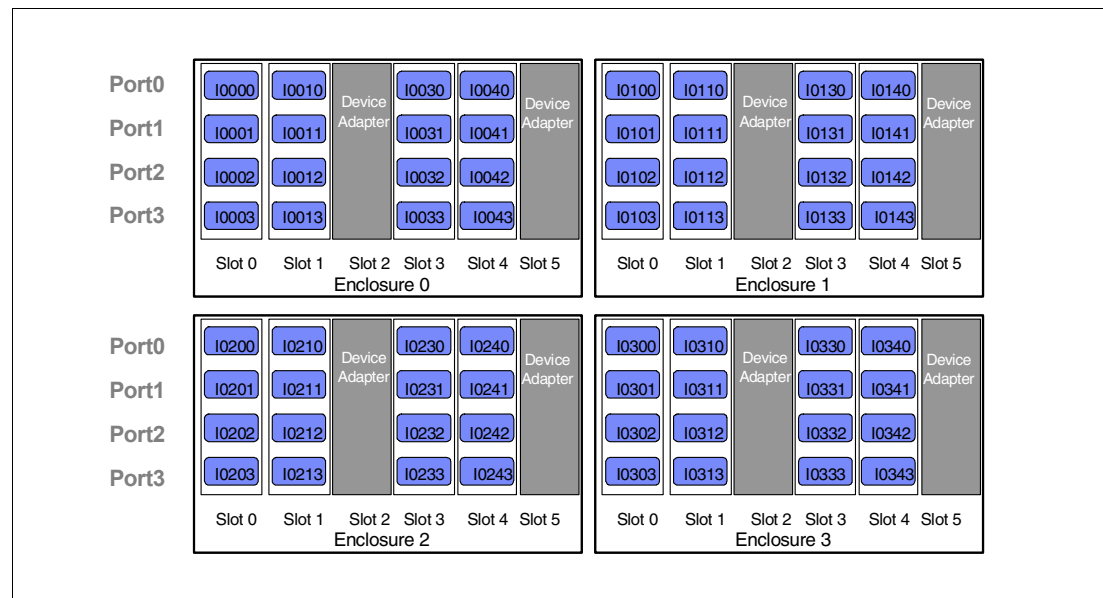


Figure 6-12 DS8000 port numbering

The I/O INTENSITY column is the result of multiplication of the operations per second and the response time per operation. For FICON ports, this information is calculated for both the read and write operations; for PPRC ports, it is calculated for both the send and receive operations. The total I/O intensity is the sum of those two numbers on each port.

For FICON ports, if the total I/O intensity reaches 4000, the response time is significantly affected, most probably, the PEND and CONN times. When this number already approaches 2000, actions might be needed to prevent further increase in the total I/O intensity. See the discussion about PEND and CONN times in “PEND time” on page 376 and “CONN time” on page 377. This rule does not apply for PPRC ports, especially if the distance between the primary site and the secondary site is significant.

### 6.3.8 Extent pool and rank

One entire rank can be assigned only to one extent pool. First, we describe the configuration where one extent pool has only one rank. Then, we describe the configuration for multiple ranks without and with Storage Pool Striping.

#### One rank on one extent pool

Example 6-6 shows the rank performance statistics with one rank per extent pool. The important metric here is the read response time per operation. If this number is greater than 35 ms, it is an indication that the DDMs within the rank are saturated. This saturation happens if too many I/O operations are executed against this rank. In this case, we need to identify the volumes on this rank that have extremely high I/O rates or extremely high (read + write) MBps throughput. Move a few of these volumes to other, less busy ranks.

*Example 6-6 Rank statistics for one rank per extent pool*

ESS RANK STATISTICS															
z/OS V1R8			SYSTEM ID SYSA				DATE 10/29/2008				INTERVAL 14.59.875				
			CONVERTED TO z/OS V1R10 RMF				TIME 02.14.00				CYCLE 0.750 SECONDS				
SERIAL NUMBER 00000ABCD1			TYPE-MODEL 002107-921				CDATE 10/29/2008				CTIME 02.14.01 CINT 14.59				
--EXTENT POOL--			----- READ OPERATIONS -----				----- WRITE OPERATIONS -----				--ARRAY--				
ID	TYPE	RRID	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	NUM	WDTH	RPM	RANK CAP TYPE	
0000	CKD	1Gb	0000	3.7	35.8K	133.6K	9.6	11.3	57.9K	656.8K	61.9	1	6	15	876G RAID 5
0001	CKD	1Gb	0001	32.9	48.3K	1.6M	4.5	11.7	35.0K	410.7K	24.1	1	6	15	876G RAID 5
0002	CKD	1Gb	0002	11.2	43.4K	484.2K	7.2	37.9	123.1K	4.7M	120.1	1	6	15	876G RAID 5
0003	CKD	1Gb	0004	57.7	49.9K	2.9M	8.8	88.1	145.2K	12.8M	126.3	1	6	15	876G RAID 5
0004	CKD	1Gb	0006	153.3	53.4K	8.2M	9.5	87.2	143.5K	12.5M	135.1	1	6	15	876G RAID 5
0005	CKD	1Gb	0007	329.8	45.5K	15.0M	5.6	28.4	156.6K	4.4M	68.3	1	6	15	876G RAID 5
0006	CKD	1Gb	0008	110.7	46.1K	5.1M	7.2	25.6	16.3K	418.3K	53.4	1	6	15	876G RAID 5
0007	CKD	1Gb	000A	149.5	54.9K	8.2M	3.9	29.3	87.2K	2.6M	88.6	1	6	15	876G RAID 5
0008	CKD	1Gb	000C	1495.4	53.5K	80.1M	58.9	140.1	182.8K	25.6M	325.5	1	6	15	876G RAID 5
0009	CKD	1Gb	000D	579.5	54.3K	31.5M	13.8	84.1	162.3K	13.6M	103.5	1	6	15	876G RAID 5

A high write response time is not a concern, because the write operation is an asynchronous operation. The actual write operation from the cache/NVS to the rank is performed after the host write I/O is considered completed, which is when the updated record is already written to both the cache and NVS.

When NVS is filled up to a high-water mark percentage, the least recently used data is written down to the rank until NVS usage is reduced to a low-water mark percentage. During this activity, multiple write requests can be queued to the same DDM, which can result in the elongation of the write response time. High write response times are not usually an indication of a performance problem.

- One LCU on one extent pool

When one LCU is assigned on one extent pool, analyzing the LCU's back-end performance is as simple as looking at the ESS RANK STATISTICS report for the extent pool/rank. See Example 6-6.

- One LCU on multiple extent pools

If an LCU uses multiple extent pools, we can still see the back-end performance of each extent pool/rank that belongs to that LCU. If a certain rank is saturated, we can identify which volumes of the LCU reside on that particular rank. See the report in Figure 6-11 on page 331 to learn which volumes are on which ranks.

- Multiple LCUs on one extent pool

Here, we have a configuration where multiple LCUs are allocated on the same extent pool and thus the same rank. If there is a rank saturation, it is difficult to determine which LCU is causing the problem. The LCU with the highest response time might just be a victim and not

the perpetrator of the problem. The perpetrator is usually the LCU that is flooding the rank with I/Os.

## Multiple ranks on one extent pool without Storage Pool Striping (SPS)

Now, we examine the effect on performance analysis when we have multiple ranks defined on one extent pool. Example 6-7 shows the rank statistics for that configuration. In this example, extent pool 0000 contains ranks with RRID 0001, 0003, 0005, 0007, 0009, 000B, 000D, 000F, 0011, 0013, and 001F. Each rank's performance statistics, and the weighted average performance of the extent pool, are reported here.

Regardless of the way in which we define the LCU in relationship to the extent pool, identifying the cause of a performance problem is complicated, because we can see only the association of a volume to an extent pool and not to a rank. See Example 6-4 on page 332. The DS CLI **showrank** command can provide a list of all of the volumes that reside on the rank that we want to investigate further. Analyzing performance based on **showrank** output can be difficult, because it can show volumes from multiple LCUs that reside on this one rank.

*Example 6-7 Rank statistics for multiple ranks on one extent pool without SPS*

ESS RANK STATISTICS														
z/OS V1R8			SYSTEM ID SYSE			DATE 09/28/2008			INTERVAL 14.59.942					
			CONVERTED TO z/OS V1R10 RMF			TIME 11.59.00			CYCLE 1.000 SECONDS					
SERIAL NUMBER	00000BCDE1		TYPE-MODEL	002107-921		CDATE	09/28/2008		CTIME	11.59.01		CINT	14.59	
--EXTENT POOL--			----- READ OPERATIONS -----				----- WRITE OPERATIONS -----				--ARRAY--			
ID	TYPE	RRID	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	NUM	WDTH	RPM	RANK CAP TYPE
0000	CKD 1Gb	0001	458.1	53.1K	24.3M	2.5	16.8	10.6K	178.5K	15.2	1	6	15	876G RAID 5
		0003	95.2	41.3K	3.9M	3.4	19.4	11.9K	230.5K	13.5	1	7	15	1022G RAID 5
		0005	580.4	54.0K	31.3M	2.7	15.8	22.8K	359.4K	16.4	1	6	15	876G RAID 5
		0007	146.6	45.5K	6.7M	5.0	14.6	14.4K	210.4K	12.6	1	7	15	1022G RAID 5
		0009	30.8	22.2K	685.7K	6.1	14.5	31.8K	462.2K	14.2	1	6	15	876G RAID 5
		000B	167.6	47.2K	7.9M	4.2	20.7	52.9K	1.1M	13.1	1	7	15	1022G RAID 5
		000D	49.2	26.1K	1.3M	5.9	12.5	12.3K	152.9K	12.1	1	6	15	876G RAID 5
		000F	255.4	53.1K	13.5M	3.1	11.8	26.8K	317.5K	15.4	1	6	15	876G RAID 5
		0011	103.0	39.3K	4.1M	7.2	20.4	21.5K	437.2K	15.0	1	7	15	1022G RAID 5
		0013	127.0	47.5K	6.0M	3.2	7.3	39.3K	285.9K	13.7	1	6	15	876G RAID 5
		001F	1.0	9.4K	9.6K	7.5	1.8	43.6K	78.7K	22.7	1	7	15	1022G RAID 5
		POOL	2014.3	49.5K	99.8M	3.4	155.5	24.5K	3.8M	14.2	11	71	15	10366G RAID 5
0001	CKD 1Gb	0000	129.4	51.3K	6.6M	2.3	6.8	21.9K	149.3K	12.5	1	6	15	876G RAID 5
		0002	228.6	50.9K	11.6M	2.0	16.7	38.9K	648.1K	14.7	1	7	15	1022G RAID 5
		0004	51.0	36.1K	1.8M	5.2	7.3	17.1K	125.8K	11.0	1	6	15	876G RAID 5
		0006	189.3	52.6K	10.0M	1.8	7.3	17.3K	126.4K	11.6	1	6	15	876G RAID 5
		0008	160.5	49.1K	7.9M	2.1	8.8	28.1K	246.5K	13.6	1	6	15	876G RAID 5
		000A	71.4	38.2K	2.7M	6.3	12.2	11.5K	140.1K	10.4	1	6	15	876G RAID 5
		000C	230.1	50.9K	11.7M	2.7	11.8	15.0K	176.4K	13.1	1	7	15	1022G RAID 5
		000E	183.0	50.6K	9.3M	2.6	16.6	86.8K	1.4M	15.1	1	6	15	876G RAID 5
		0010	150.1	50.3K	7.6M	2.6	6.2	22.0K	136.2K	12.5	1	7	15	1022G RAID 5
		0012	193.2	51.6K	10.0M	4.4	7.6	24.1K	182.8K	14.3	1	7	15	1022G RAID 5
		001E	4.5	49.8K	224.8K	16.6	0.6	57.0K	32.7K	83.1	1	7	15	1022G RAID 5
		POOL	1591.2	49.9K	79.4M	2.9	101.8	33.4K	3.4M	13.5	11	71	15	10366G RAID 5

## Multiple ranks on one extent pool using Storage Pool Striping

The same performance analysis challenges still exist either when we use the SPS or non-SPS technique, because in both cases there are multiple volumes from multiple LCUs that are allocated on each rank within the extent pool.

The benefit of using SPS is that the probability of rank saturation is significantly reduced compared to not using SPS. Example 6-8 on page 336 shows extent pool 0000, which contains RRID 0000, 0002, 0008, 000A, 0010, 0012, 0018, and 001A. This report shows a balanced load on these ranks. All of the performance metrics, such as OPS/SEC, BYTES/OP, BYTES/SEC, and RTIME/OP, are balanced across all ranks that reside on this extent pool for the read and also the write operations.

Compare Example 6-8 to Example 6-7 on page 335. In Example 6-7 on page 335, we see that the OPS/SEC and the BYTES/SEC vary widely between the ranks within an extent pool, which can cause certain extent pools to reach saturation; other extent pools do not. If we had used SPS in the previous configuration example, the possibility of rank saturation is reduced significantly.

**Note:** Regarding the advantages of SPS, consider it as the first choice in configuring a new DS8000, unless there is a compelling reason not to use it.

Example 6-8 Rank statistics for multiple ranks on one extent pool using SPS

ESS RANK STATISTICS															
z/OS V1R9			SYSTEM ID SYS5				DATE 11/05/2008				INTERVAL 01.00.003				
			CONVERTED TO z/OS V1R10 RMF				TIME 21.24.00				CYCLE 0.333 SECONDS				
SERIAL	NUMBER	00000AB321	TYPE-MODEL 002107-932				CDATE	11/05/2008	CTIME	21.24.00	CINT	01.00			
--EXTENT POOL--			----- READ OPERATIONS -----				----- WRITE OPERATIONS -----				--ARRAY--		MIN	RANK	RAID
ID	TYPE	RRID	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	OPS /SEC	BYTES /OP	BYTES /SEC	RTIME /OP	NUM	WDTH	RPM	CAP	TYPE
0000	CKD 1Gb	0000	40.9	49.1K	2.0M	6.4	11.3	13.1K	148.5K	7.1	1	6	15	438G	RAID 5
		0002	39.2	47.6K	1.9M	6.8	11.9	15.8K	187.9K	7.5	1	6	15	438G	RAID 5
		0008	40.9	48.2K	2.0M	6.4	10.9	16.5K	179.1K	7.2	1	6	15	438G	RAID 5
		000A	37.7	48.4K	1.8M	6.5	10.3	13.8K	142.0K	7.2	1	6	15	438G	RAID 5
		0010	40.9	48.9K	2.0M	6.2	11.4	14.4K	163.8K	7.3	1	6	15	438G	RAID 5
		0012	40.0	47.8K	1.9M	6.2	10.5	12.3K	128.9K	7.1	1	6	15	438G	RAID 5
		0018	42.1	47.8K	2.0M	6.6	12.1	12.4K	150.7K	7.2	1	6	15	438G	RAID 5
		001A	45.9	48.6K	2.2M	6.7	12.7	15.1K	192.2K	7.3	1	6	15	438G	RAID 5
		POOL	327.8	48.3K	15.8M	6.5	91.1	14.2K	1.3M	7.2	8	48	15	3504G	RAID 5
0001	CKD 1Gb	0004	40.8	47.8K	1.9M	6.7	11.2	14.0K	157.3K	9.2	1	7	15	511G	RAID 5
		0006	42.0	48.8K	2.0M	6.9	11.9	16.1K	192.2K	9.2	1	7	15	511G	RAID 5
		000C	38.4	49.4K	1.9M	6.5	9.5	16.5K	157.3K	8.8	1	7	15	511G	RAID 5
		000E	43.0	49.0K	2.1M	6.6	12.1	16.6K	201.0K	8.5	1	7	15	511G	RAID 5
		0014	40.4	49.2K	2.0M	6.6	10.8	15.2K	163.8K	8.4	1	7	15	511G	RAID 5
		0016	40.5	48.8K	2.0M	6.5	11.5	16.0K	183.5K	8.5	1	7	15	511G	RAID 5
		001C	37.9	47.0K	1.8M	6.2	10.5	14.5K	152.9K	8.2	1	7	15	511G	RAID 5
		001E	39.6	48.6K	1.9M	7.2	10.9	16.5K	179.1K	10.0	1	7	15	511G	RAID 5
		POOL	322.7	48.6K	15.7M	6.7	88.4	15.7K	1.4M	8.9	8	56	15	4088G	RAID 5

## 6.4 RMF Magic for Windows

RMF Magic for Windows® is a tool available from IntelliMagic B.V., a company that specializes in storage performance and modeling software. Information for obtaining this product is available through the IntelliMagic web site at:

<http://www.intellimagic.net>

RMF Magic provides consolidated performance reporting about your z/OS disk subsystems from the point of view of those disk subsystems, rather than from the host perspective, even when disk subsystems are shared between multiple sysplexes. This disk-centric approach makes analyzing the I/O configuration and performance much easier. RMF Magic automatically determines the I/O configuration from your RMF data, showing the relationship between the disk subsystem serial numbers, SSID, LCUs, device numbers, and device types. With RMF Magic, there is no need to manually consolidate printed RMF reports.

Although RMF Magic reports are based on information from RMF records, the analysis and reporting goes beyond what RMF provides. In particular, the tool computes accurate estimates for the read and write bandwidth (MBps) for each disk subsystem and down to the device level. With this unique capability, RMF Magic can size the links in a future remote copy configuration, because RMF Magic knows the bandwidth that is required for the links, both in I/O requests and in megabytes per second (MBps), for each point in time.

RMF Magic consolidates the information from RMF records with channel, disk, LCU, and cache information into one view per disk subsystem, per SSID (LCU), and per storage group. RMF Magic gives insight into your performance and workload data for each RMF interval within a period selected for the analysis, which can span weeks. Where RMF postprocessor reports are sorted by host and LCU, RMF Magic reports are sorted by disk subsystem and SSID (LSS). With this information, you can plan migrations and consolidations more effectively, because RMF Magic provides a detailed insight in the workload, both from a disk subsystem and a storage group perspective.

RMF Magic's graphical capabilities allow you to find any hot spots and tuning opportunities in your disk configuration. Based on user-defined criteria, RMF Magic can automatically identify peaks within the analysis period. In addition, the graphical reports make all of the peaks and anomalies evident immediately, which helps you with analyzing and identifying performance bottlenecks.

You can use RMF Magic to analyze subsystem performance for z/OS hosts. If your DS8000 also provides storage for Open Systems hosts, the tool provides reports on rank statistics and host port link statistics. The DS8000 storage subsystem provides these Open Systems statistics to RMF when performance data is reported to RMF and is available for reporting through RMF Magic. Of course, if you have a DS8000 that has only Open Systems activity and does not include any z/OS 3390 volumes, this data cannot be collected by RMF and reported on by RMF Magic.







## DB2 I/O operations

System z synergy is one of the key factors in improving performance. DB2 uses the latest improvements in hardware and operating system to provide better performance, improved value, more resilience, and better function. Faster fiber channels and improved IBM System Storage DS8000 performance provide faster data access, and DB2 9 makes adjustments to improve I/O performance more. FlashCopy can be used for DB2 backup and restore. DB2 makes unique use of the z/Architecture® instruction set, which has new long-displacement instructions and better performance for them on the latest processors. DB2 continues to deliver synergy with data and index compression. The z/OS Workload Manager (WLM) improvements will help manage DB2 buffer pools.

The information described here might be similar to, or shared with, parts of *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840. This information is provided to give storage administrators an understanding of the I/O operations performed by DB2.

- ▶ Overview of I/O options
- ▶ Data read operations
- ▶ Data write operations
- ▶ Log writes
- ▶ Log reads
- ▶ Distributing data sets more efficiently
- ▶ DB2 sort work files
- ▶ DB2 and tape processing

## 7.1 Overview of I/O options

The two most important I/O operations performed by DB2 are the data read I/O and the log write I/O. The data read I/O has direct impact on the response time of any SQL query. The log write I/O has an important impact on online transaction response time. This chapter describes the following I/O operations in detail:

- ▶ Data I/O: read and write accesses to DB2 table spaces and index spaces
- ▶ Log I/O: read and write accesses to the active log.

Other I/O operations are performed by DB2, like access to image copies, archive logs, and BSDSs. These I/O operations are not described in this chapter.

### 7.1.1 Avoiding I/O operations

Before you analyze DB2 for performance problems, look at the overall system. If the initial analysis suggests that the performance problem is within DB2, the problem might be poor response time, an unexpected and unexplained high use of resources, or locking conflicts. Check factors such as total processor usage, disk activity, and paging.

One of the basic principles of DB2 design is to avoid I/O operations if possible. DB2 tries to achieve this, by using a hierarchy of buffer pools to keep data in memory. Modern disk devices complement this by having large caches which give an additional level of intermediate data storage. The storage hierarchy is illustrated in Figure 7-1 on page 342.

DB2 uses virtual buffer pools to store the data pages. I/O intensive virtual buffer pools could optionally fully backed up by real storage using the PAGEFIX YES buffer pool parameter. This can reduce CPU utilization by increasing the DB2 allocation of real storage for that LPAR.

For buffer pool sizing, see the following publications:

- ▶ Section “4.5 Buffer pool long term page fixing” of *DB2 UDB for z/OS Version 8 Performance Topics*, SG24-6465
- ▶ *DB2 9 for z/OS: Buffer Pool Monitoring and Tuning*, REDP-4604

When data sharing is used, group buffer pools in the coupling facility store updated pages before these are cast out to disk. Pages in the group buffer pool can be accessed from any member of the data sharing group.

In addition to the caching done by DB2, the storage controller also uses a cache for data. The controller has algorithms to determine the convenience of pre-staging the data to the cache. For example, if several sequential reads are detected, the controller reads tracks ahead of the requests to improve the cache hit ratio.

### 7.1.2 How I/O is performed in DB2

DB2 can schedule four types of read I/O for SQL calls:

- ▶ Synchronous I/O of a single page, also called a random read
- ▶ Sequential prefetch of up to 32 (DB2 V8) or 64 (DB2 9) contiguous 4 KB pages
- ▶ List prefetch of up to 32 contiguous or non-contiguous 4 KB pages
- ▶ Dynamic prefetch of up to 32 contiguous 4 KB pages

Note that the prefetch quantity for utilities can be twice that for SQL calls. A synchronous I/O is scheduled when a requested page is not found in the local or, if data sharing, global buffer pool and always results in a delay to the application process whilst the I/O is performed. A prefetch I/O is triggered by the start of a sequential or list prefetch getpage process or by dynamic sequential detection. The I/O is executed ahead of the getpage request, apart from the initial pages, for a set of pages which are contiguous for sequential and dynamic prefetch.

List prefetch normally results in a non-contiguous sequence of pages, however these pages can also be contiguous. The I/O is executed by a separate task, referred to as a *prefetch read engine*, which first checks whether any of the pages in the prefetch request are already resident in the local pool.

If they are, they are then removed from the I/O request. Sequential prefetch does not start until a sequential getpage is not found in the pool, referred to as a *page miss*. When triggered, it continues the prefetch process irrespective of whether the remaining pages are already resident in the pool. List and dynamic prefetch engines are started whether or not the pages are already in the pool. As a result prefetch engines can be started only to find that all pages are resident in the pool. The prefetch engines can be disabled at the buffer pool level by setting VPSEQT=0. There is a limit, dependent on the version of DB2, on the total number of read engines. The DB2 Statistics report shows when prefetch has been disabled because of the limit being reached. The actual number of pages fetched depends on the buffer pool page size, the type of prefetch, the size of the buffer pool, and the release of DB2.

For more information about the specification, see the following publications:

- For V8, see *DB2 UDB for z/OS Version 8 Administration Guide*, SC18-7413
- For DB2 9, see *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851.

Dynamic prefetch is triggered by a sequence of synchronous getpages, which are close to sequential and can be either ascending or descending in direction. If a page is not in the virtual pool, which includes the group buffer pool for data sharing, DB2 schedules a synchronous I/O, irrespective of the type of getpage. For this reason, the first one or two pages of a sequential prefetch are read with synchronous I/Os.

In DB2 9, sequential prefetch is used only for table space scans. Other access paths that previously used sequential prefetch, now rely on dynamic prefetch. The result of this change is more use of dynamic prefetch I/O than in previous versions of DB2. This change in behavior takes place in DB2 9 conversion mode and is automatic with no REBIND required.

The hierarchy shown in Figure 7-1 on page 342 depicts a typical DB2 storage and I/O usage. For more details, see *DB2 9 for z/OS Performance Topics*, SG24-7473.

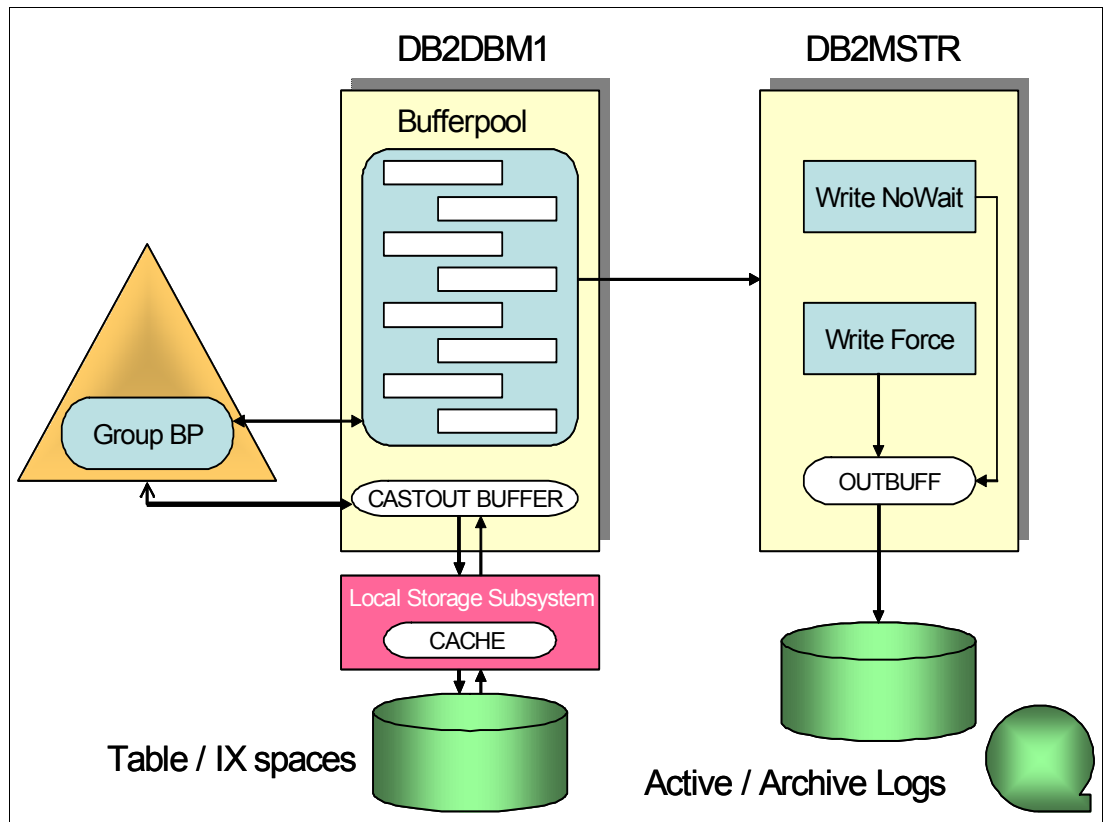


Figure 7-1 Diagram of DB2 storage

## 7.2 Data read operations

DB2 sequential I/O is important in the following applications:

- ▶ Daily full image copy cycles
- ▶ Processing of LOBs (large objects)
- ▶ Daily or weekly table unloads to feed data warehouses or marts
- ▶ Database recovery

In addition, DB2 uses four read mechanisms to get data pages from disk into the virtual buffer pool:

- ▶ Normal read (or synchronous read)
- ▶ Sequential prefetch
- ▶ Dynamic prefetch
- ▶ List sequential prefetch

### 7.2.1 Normal read

Normal read mechanism is used when only one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page. This read mechanism is referred to in OMEGAMON PE reports as *synchronous read*. See **A** in Example 7-1 on page 354.

## 7.2.2 Sequential prefetch

When the optimizer chooses sequential prefetch as the access path, sequential prefetch is performed concurrently with other operations of the originating application program. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Because this read mechanism executes concurrently and independently of the application program, it is referred to in OMEGAMON PE reports as *asynchronous read*. See **B** in Example 7-3 on page 356. Of course, not all asynchronous I/O can be performed concurrently; certain instances do not have total overlap, in which wait times will still appear in the accounting records.

Sequential prefetch can be used to read data pages, by table space scans or index scans with clustered data reference. It can also be used to read index pages in an index scan. Sequential prefetch allows CP and I/O operations to be overlapped.

Because sequential prefetch reads multiple pages in one I/O operation, it has an important performance advantage over the normal read for applications that process multiple sequential pages. The DB2 virtual buffer pools must therefore be large enough to avoid situations in which prefetched pages are being stolen by another application before they are referenced.

The chart in Figure 7-2 depicts how the I/O hardware has been improving geometrically for the last several years, starting with the 3990-6 control unit in 1993. DB2 Prefetch I/Os are used for illustration, but the same improvements were more or less true for all sequential I/Os. The RVA doubled the throughput over the 3990-6, but the real improvement occurred in the first two years of the new millennium, starting with the first shipment of ESS DASD, known then as the Model E20 in the year 2000. ESS DASD doubled the throughput compared to RVA, but within two years, by 2002 (ESS800) throughput had quadrupled compared to that first ESS. In 2001 IBM delivered the F20 model, which ushered in support for FICON channels, starting with 1 Gb and the 2 Gbps channels. In the z900 processor, IBM included FICON Express channels.

Note that on FICON channels, non-Extended Format (non-EF) performance was beginning to exceed the performance of Extended Format (EF) data sets. The performance improves with every new device. The rate of improvement for non-EF data sets was lower compared to EF data sets.

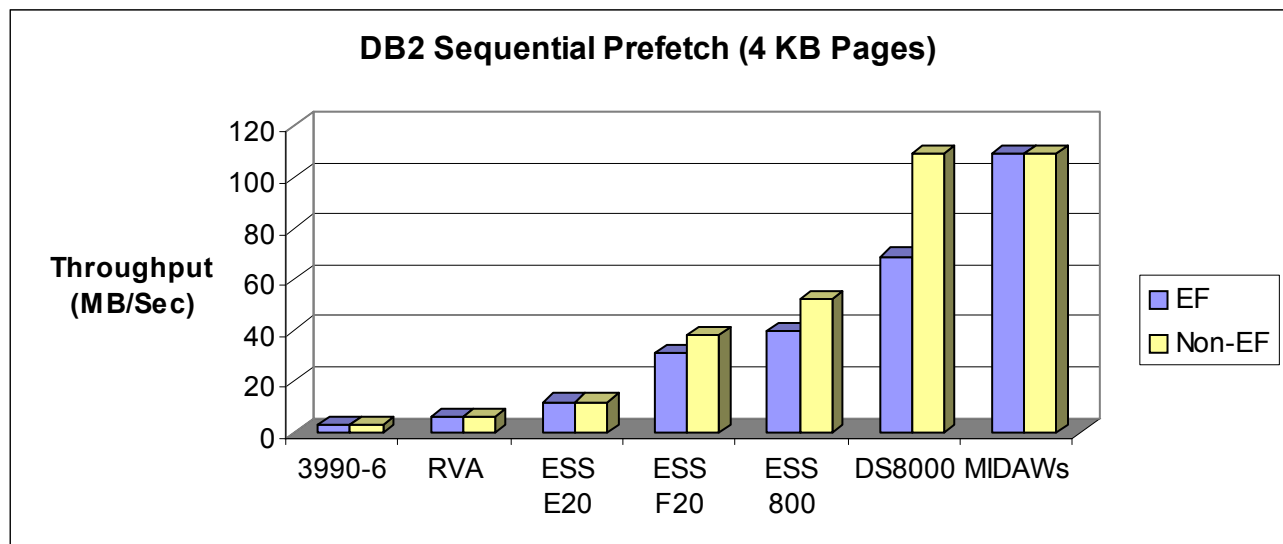


Figure 7-2 DB2 sequential prefetch

DB2 can process, for the most part, sequentially accessed pages in the DB2 buffer pool faster than the I/O subsystem can satisfy DB2's request for another page. Reading several consecutive pages into the buffer pool using a single I/O operation can greatly reduce the overhead associated with running your application. In addition, performing multiple I/O operations in parallel to read in several ranges of pages at the same time can help reduce the time that your application must wait for I/O operations to complete.

After DB2 is made aware that this sequential access might happen, DB2 might then choose sequential prefetch as the preferred access path. DB2's sequential prefetch is an asynchronous process that reads a number of pages into the buffer pools before DB2 actually needs to use them. The number of pages read is usually based on the buffer pool size. (More about buffer pools is described in the following sections.)

Each prefetch read occurs when a *trigger page* is reached. The trigger page also is determined by the number of pages read in each prefetch operation. The idea is to stage the pages that DB2 (actually DB2's buffer manager component) may possibly need in the pool before DB2 actually requires them. Remember that DB2 "thinks" it is sequentially reading these pages. This read-ahead processing can reduce the chances of DB2 having to wait for a page I/O to complete before satisfying a getpage request

### 7.2.3 Dynamic prefetch

Standard sequential prefetch is established at BIND time, when the optimizer establishes a sequential access path to the data. At execution time however, DB2 Data Manager may choose dynamic prefetch. DB2 uses a *sequential detection* algorithm to determine that pages are accessed in a sequential access pattern, and which activates sequential prefetch. This type of sequential prefetch is called dynamic prefetch. An algorithm is also used to disable the dynamic prefetch.

Sequential prefetch schedules one SRB at a time (except at the start of a query); dynamic prefetch schedules two concurrent SRBs. This scheduling in turn can double throughput in the following situations:

- ▶ The data is cache-resident.
- ▶ The channels are not busy.
- ▶ The scan has low CPU time.

Table space scans always use sequential prefetch. For indexes in DB2 Version 8, the DB2 optimizer made the choice based on index statistics. In Version 9, the optimizer always relies on dynamic prefetch for everything other than table space scans.

Dynamic prefetch occurs when the optimizer establishes a non-sequential access path to the data (for example: `SELECT... WHERE KEY = :variable`); and the keys that are provided to the program are many, and in sequential or nearly sequential order. Dynamic prefetch provides the same advantages as sequential prefetch.

As mentioned, dynamic prefetch is based on sequential detection at run time and is based on access pattern. Sequential prefetch does not provide support for backward prefetch; dynamic prefetch provides support for both forward and backward directions.

Dynamic prefetch can switch automatically between multi-page prefetch read and single-page synchronous read, as needed. This process is effective when the cluster ratio of index or indexes that are used prior to data access is less than 100%.

Notice that DB2 9 for z/OS has also improved the way the data clustering information is collected by RUNSTATS. By switching to dynamic prefetch, DB2 9 avoids wasteful prefetch I/Os for disorganized indexes and for skip sequential index and data page access.

The dynamic prefetch enhancement is applicable to single table access, multi-table join, outer join, subquery, and union. This enhancement affects the prefetch operations during index scan or table access through index scan in an SQL call. Utilities already used dynamic prefetch.

Lab measurements indicate a query performance with an elapsed time improvement of 5 - 50% for queries that use index scan. Up to a 10% reduction of synchronous I/O for data pages and a possible reduction of up to 75% of synchronous I/O for index pages can occur. There is some reduction in CPU time.

Dynamic prefetch requests are detailed in OMEGAMON PE reports. For an example, see **D** in Example 7-4 on page 364.

## 7.2.4 List prefetch

List prefetch is used to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch reads a set of data pages that are determined by a list of RIDs taken from an index. The data pages need not be contiguous. The maximum number of pages that can be retrieved in a single list prefetch is 32 (64 for utilities). Before the read is performed, the RIDs are sorted in sequential order, allowing clustered accesses. List prefetch can also be used by incremental image copy

List prefetch can be used in conjunction with either single or multiple index access.

List prefetch uses any of the following three steps:

- ▶ RID retrieval: A list of RIDs for needed data pages is found by matching index scans of one or more indexes.
- ▶ RID sort: The list of RIDs is sorted in ascending order by page number.
- ▶ Data retrieval: The needed data pages are prefetched in order using the sorted RID list.

List prefetch does not preserve the data ordering given by the index. Because the RIDs are sorted in page number order before accessing the data, the data is not retrieved in order by any column. If the data must be ordered for an ORDER BY clause or any other reason, it requires an additional sort.

List prefetch is used under the following circumstances:

- ▶ Usually with a single index that has a cluster ratio lower than 80%
- ▶ Sometimes on indexes with a high cluster ratio, if the estimated amount of data to be accessed is too small to make sequential prefetch efficient, but large enough to require more than one regular read
- ▶ Always to access data by multiple index access
- ▶ Always to access data from the inner table during a hybrid join
- ▶ Usually for updatable cursors when the index contains columns that might be updated

List prefetch requests are detailed in OMEGAMON PE reports. For an example, see **C** in Example 7-4 on page 364.

## 7.2.5 Prefetch quantity

The sequential, dynamic, and list prefetch operations each read a set of pages. The maximum number of pages read by a request issued from an application program is determined by the size of the buffer pool used.

DB2 9 for z/OS conversion mode uses dynamic prefetch for index scans instead of using sequential prefetch. Dynamic prefetch is more intelligent and more robust.

Sequential prefetch is still used for table space scans and has been enhanced in DB2 9 for z/OS. Larger prefetch quantity and deferred write quantity are used in DB2 9 for large buffer pools, which are defined as follows:

- ▶ For sequential prefetch, if VPSEQT multiplied by VPSIZE is greater than 160 MB for SQL, 320 MB for utility
- ▶ For deferred write, if VPSIZE is greater than 160 MB for SQL, 320 MB for utility

The maximum prefetch quantity goes from 128 KB (with V8) to 256 KB (with DB2 9) in SQL for a table space scan. It goes from 256 KB (with V8) to 512 KB (with DB2 9) in a utility.

**Note:** PREFETCH column: The PREFETCH column of the PLAN\_TABLE still shows the value of “S” for sequential prefetch, even though DB2 uses dynamic prefetch for index scans.

When a virtual buffer pool is very small, sequential prefetch is disabled. Prefetch is also disabled if the sequential prefetch thresholds is reached. This is explained in 7.2.7, “Sequential prefetch threshold” on page 348.

The number of pages that are read by prefetch method depends on the type of prefetch, the buffer pool size (VPSIZE), the sequential steal threshold (VPSEQT), and the number of buffers.



Table 7-1, shows the number pages that prefetch reads for each asynchronous I/O for each buffer pool size (4 KB, 8 KB, 16 KB, and 32 KB).

*Table 7-1 The number of pages read by prefetch by buffer pool size*

Buffer pool size	Number of buffers: VPSIZE	Sequential and LOB list	Dynamic and non-LOB list	Utility sequential
4 KB	< 224	8	8	16
	225 to < 1,000	16	16	32
	<= 1000 to < 40,000 or VPSIZE*VPSEQT < 40 000	32	32	64
	<= 40,000 VPSIZE*VPSEQT and < 80,000	64	3232	64
	80,000 <= VPSIZE*VPSEQT	64		128
8 KB	< 48	4	4	8
	48 to <400	8	8	16
	<= 400 to < 20,000 or VPSIZE*VPSEQT < 20,000	16	16	32
	<= 20,000 to VPSIZE*VPSEQT < 40,000	32	16	32
	40,000 <= VPSIZE*VPSEQT	32	16	64
16 KB	< 24	2	2	4
	24 to < 200	4	4	8
	200 <= VPSIZE < 10,000 or VPSIZE*VPSEQT < 10,000	8	8	16
	10,000 <= VPSIZE*VPSEQT < 20,000	16	8	16
	20,000 <= VPSIZE*VPSEQT	16	8	32
32 KB	VPSIZE < 12	1	1	2
	12 < VPSIZE < 100	2	2	4
	100 <= VPSIZE < 5,000 or VPSIZE*VPSEQT < 5,000	4	4	8
	5,000 <= VPSIZE*VPSEQT < 10,000	8	4	8
	10,000 <= VPSIZE*VPSEQT	8	4	17

Table 7-3 on page 350 shows the prefetch quantity as a function of the page size and the buffer pool size. For certain utilities (REORG, RECOVER), the prefetch quantity can be twice as much.

From the OMEGAMON PE accounting trace, the average number of pages read in prefetch operations by an application program can be calculated. The average number of pages read is the total number of pages read in prefetch operations (**E** in Table 7-6 on page 353) divided by the sum of prefetch operations (**B**, **C**, **D** in Table 7-6 on page 353), which is the average pages read by one prefetch operation =  $E / (B+C+D)$ .

## 7.2.6 Data manager threshold

Fixed and user-controlled thresholds influence how DB2 uses the buffer pools. Several of these thresholds are governed by certain preset values. Each threshold is a level of use that, when exceeded, causes DB2 to take an action. Certain thresholds might indicate a buffer pool shortage problem; other thresholds merely report normal buffer management by DB2. The level of use is usually expressed as a percentage of the total size of the buffer pool. For example, the *immediate write threshold* of a buffer pool is set at 97.5%. When the percentage of unavailable pages in a buffer pool exceeds that value, DB2 writes pages to disk when updates are completed.

For very small buffer pools, of fewer than 1000 buffers, some of the thresholds might be lower to prevent *buffer pool full* conditions, but those thresholds are not described in this book.

The data manager threshold (DMTH), sometimes referred to as the buffer critical threshold, occurs when 95% of all buffer pages are unavailable (in use). The DMTH is maintained and checked independently for each individual buffer pool.

This threshold is checked before a page is read or updated. If the threshold has not been exceeded, DB2 accesses the page in the virtual buffer pool once for each page, no matter how many rows are retrieved or updated in that page. If the threshold has been exceeded, DB2 accesses the page in the virtual buffer pool once for each row that is retrieved or updated in that page. Reaching this threshold has a significant effect on processor usage and performance.

The DB2 buffer manager requests that all threads to release any possible pages immediately. This step occurs if you set GETPAGE/RELPAGE processing by row instead of page. After a GETPAGE is executed and a single row is processed, a RELPAGE request is issued, which causes CPU to become high for objects in the buffer pool, and transactions that are I/O-sensitive can suffer. This result can occur if the buffer pool is too small.

You can observe when this happens by a non-zero value in the DM THRESHOLD REACHED indicator on a statistics report. This threshold is checked every time a page is read or updated. If it is not reached, DB2 accesses the page in the virtual pool once for each page (no matter how many rows are used). If the threshold has been reached, DB2 accesses the page in the virtual pool once for every row on the page that is retrieved or updated, which can lead to serious performance degradation.

**Note:** Avoid reaching the DMTH because it has a significant effect on processor usage. The DMTH is maintained for each individual buffer pool. When the DMTH is reached in one buffer pool, DB2 does not release pages from other buffer pools

## 7.2.7 Sequential prefetch threshold

The sequential prefetch threshold (SPTH) is checked before a prefetch operation is scheduled and during buffer allocation for a previously scheduled prefetch. If the SPTH is exceeded, prefetch either is not scheduled or is canceled. The SPTH is set by DB2 at 90% of each virtual buffer pool. This threshold is checked at two times:

- Before scheduling a prefetch operation. If the threshold has been exceeded, the prefetch is not scheduled.
- During buffer allocation for an already-scheduled prefetch operation. If the threshold has been exceeded, the prefetch is canceled.

When the SPTH is reached, sequential prefetch is disabled until more buffers become available, which adversely affects the performance of operations that use sequential prefetch.

If you look at the Statistics Report, the indicator PREFETCH DISABLED -NO BUFFER is incremented every time a virtual buffer pool reaches 90% of active unavailable pages, disabling sequential prefetching. This value must always be zero; a non-zero value is a clear indication that you are probably experiencing degraded performance as a result of all prefetch being disabled.

## 7.3 Data write operations

In this section, we examine the DB2 data write operations.

### Buffer pool writes

When an application updates data, the updated pages are kept in the virtual buffer pool. Eventually, the updated data pages in the virtual buffer pool have to be written to disk. Write operations can be either asynchronous or synchronous, with the execution of the unit of work.

Write operations are usually performed concurrently with user requests. Updated pages are queued by data set until they are written when one of the following events occurs:

- ▶ A checkpoint is taken.
- ▶ The percentage of updated pages in a buffer pool for a single data set exceeds a preset limit called the vertical deferred write threshold (VDWQT).
- ▶ The percentage of unavailable pages in a buffer pool exceeds a preset limit called the deferred write threshold (DWQT).

Table 7-2 lists the number of pages DB2 can write in a single I/O operation.

*Table 7-2 Number of pages that DB2 can write in a single I/O operation*

Page size	Number of pages
4 KB	32
8 KB	16
16 KB	8
32 KB	4

Table 7-3 on page 350 lists the number of pages DB2 can write in a single utility I/O operation. If the number of buffers is large enough, DB2 can write twice as many pages for each I/O operation for a *utility-write*.

Table 7-3 Number of pages that DB2 can write to a single I/O operation for utility-writes

Page size	Number of buffers	Number of pages
4 KB	BP > 80000 pages BP < 80000 pages	128 64
8 KB	BP > 40000 pages BP < 40000 pages	64 32
16 KB	BP > 20000 pages BP < 20000 pages	32 16
32 KB	BP > 10000 pages BP < 10000 pages	16 8

As with utility-write operations, DB2 can write twice as many pages for each I/O in a LOB write operation. Table 7-4 shows the number of pages that DB2 can write for each I/O operation for a LOB write.

Table 7-4 The number of pages that DB2 can write for a single I/O operation for **LOB** writes

Page size	Number of buffers	Number of pages
4 KB	BP > 80000 pages BP < 80000 pages	64 32
8 KB	BP > 40000 pages BP < 40000 pages	32 16
16 KB	BP > 20000 pages BP < 20000 pages	16 8
32 KB	BP > 10000 pages BP < 10000 pages	8 4

## Synchronous reads and writes

*Synchronous reads* are physical pages that are read-in, one page per I/O. *Synchronous writes* are pages written one page per I/O. Consider limiting synchronous reads and writes to only what is truly necessary, which is usually small in occurrence and number. Otherwise, you might start to see buffer pool stress (too many checkpoints, as an example). DB2 begins to use synchronous writes if the immediate write threshold (IWTH) is reached or if two system checkpoints pass without a page being written that has been updated and not yet committed.

## Asynchronous read and writes

*Asynchronous reads* are several pages that are read per I/O for prefetch operations such as sequential prefetch, dynamic prefetch, or list prefetch. *Asynchronous writes* are several pages written per I/O for operations such as deferred writes.

## Events which trigger a DB2 write

DB2 externalizes pages to disk when any of the following events occur:

- ▶ The DWQT threshold is reached.
- ▶ The VDWQT threshold is reached.
- ▶ The Data set is physically closed or switched from R/W to R/O.
- ▶ The DB2 takes a checkpoint (LOGLOAD or CHKTIME is reached).
- ▶ A QUIESCE (WRITE YES) utility is executed.

Consider controlling page externalization with the DWQT and VDWQT on the ALTER statement for best performance and to avoid surges in I/O. Do not have page externalization be controlled by DB2 system checkpoints because too many pages would be written to disk at one time, causing I/O queuing delays, increased response time, and I/O spikes. During a checkpoint, all updated pages in the buffer pools are externalized to disk, and the checkpoint is recorded in the log (except for the work files).

### 7.3.1 Asynchronous writes

Most DB2 writes are done asynchronously from the application program and chained when possible. This approach helps performance and implies that the application might have long since finished, by the time its data updates are written to disk. Updated pages are kept in the virtual buffer pool for possible reuse. The reuse ratio can be obtained from the OMEGAMON PE statistics report.

Updated pages are written asynchronously when any of the following events occur:

- ▶ A checkpoint is taken, which happens when the following conditions occur:
  - The DB2 parameter LOGLOAD or CHKTIME limit is reached.
  - An active log is switched.
  - The DB2 subsystem stops executing for MODE(QUIESCE) shutdown or -STOP DB2,MODE(QUIESCE).
- ▶ The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the VDWQT.
- ▶ The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the DWQT.

Because these operations are independent from the application program, the DB2 accounting trace cannot show these writes. The OMEGAMON statistics report is required to “see” the asynchronous writes, as shown in Example 7-2 on page 355

### 7.3.2 Synchronous writes

Synchronous writes occur exceptionally when any of the following conditions occur:

- ▶ The virtual buffer pool is too small and the IWTH is exceeded. See 7.3.3, “Immediate write threshold (IWTH)” on page 352.
- ▶ More than two DB2 checkpoints have been taken during the execution of a unit of work, and an updated page has not been written out to disk.

When the conditions for synchronous write occur, the updated page is written to disk as soon as the update completes. The write is synchronous with the application program SQL request; that is, the application program waits until the write has been completed.

Sometimes DB2 uses synchronous writes even when the IWTH has not been exceeded. For example, when more than two checkpoints pass without a page being written, DB2 uses synchronous writes. Situations such as these do not indicate a buffer shortage.

These writes are shown in the OMEGAMON statistics report (See **F** and **G**) in Example 7-2 on page 355.

### 7.3.3 Immediate write threshold (IWITH)

The IWITH is set when 97.5% of all pages in the virtual buffer pool are unavailable, and cannot be changed. Monitoring buffer pool usage includes checking how often this threshold is reached. Generally, set the virtual buffer pool sizes large enough to prevent reaching this threshold.

Reaching this threshold has a significant effect on processor usage and I/O resource consumption. For example, updating three rows per page in 10 sequential pages ordinarily requires one or two asynchronous write operations. When IWITH is exceeded, the updates require 30 synchronous writes.

### 7.3.4 Deferred write threshold (DWQT)

The DWQT is a percentage of the buffer pool that might be occupied by unavailable pages, including both updated pages and in-use pages. The default value for this threshold is 30%. You can change that to any value in the range 0 - 90% by using the DWQT option on the ALTER BUFFERPOOL command.

DB2 checks this threshold when an update to a page is completed. If the percentage of unavailable pages in the buffer pool exceeds the threshold, write operations are scheduled for enough data sets (at up to 128 pages per data set) to decrease the number of unavailable buffers to 10% below the threshold.

For example, if the threshold is 50%, the number of unavailable buffers is reduced to 40%.

When the deferred write threshold is reached, the data sets with the oldest updated pages are written asynchronously. DB2 continues writing pages until the ratio goes below the threshold.

#### Vertical deferred write threshold (VDWQT)

The VDWQT is similar to the DWQT, but it applies to the number of updated pages for a single page set in the buffer pool. If the percentage or number of updated pages for the data set exceeds the threshold, writes are scheduled for that data set, up to 128 pages.

You can specify this threshold in one of two ways:

- Percentage

This way is the percentage of the buffer pool that might be occupied by updated pages from a single page set. The default value for this threshold is 5%. You can change the percentage to any value from 0% to 90%.

- Absolute number

This way is the total number of buffers in the buffer pools that might be occupied by updated pages from a single page set. You can specify the number of buffers in the range of 0 - 999. If you want to use the number of buffers as your threshold, you must set the percentage threshold to 0.

You can change the percent or number of buffers by using the VDWQT keyword on the ALTER BUFFERPOOL command. Because any buffers that count toward VDWQT also count toward DWQT, setting the VDWQT percentage higher than DWQT has no effect: DWQT is reached first, write operations are scheduled, and VDWQT is never reached. Therefore, the ALTER BUFFERPOOL command does not allow you to set the VDWQT percentage to a value greater than DWQT. You can specify a number of buffers for VDWQT that is higher than DWQT, but again, with no effect.

### ***VDWQT is overridden***

VDWQT is overridden by certain DB2 utilities, which use a constant limit of four pages rather than a percentage of the buffer pool size. LOAD, REORG, and RECOVER use a constant limit of 128 pages.

### ***VDWQT is set to 0***

If you set VDWQT to zero, DB2 implicitly uses the smaller of 1% of the buffer pool (a specific number of pages), or the number determined by the buffer pool page size as shown in Table 7-5, to avoid synchronous writes to disk.

*Table 7-5 Number of changes pages based on buffer pool size*

Buffer pool page size	Number of changed pages
4 KB	40
8 KB	24
16 KB	16
32 KB	12

## **7.3.5 Write quantity**

DB2 writes a variable number of pages in each I/O operation. Table 7-6 shows the maximum pages that DB2 can write in a single asynchronous I/O operation. Certain utilities can write twice the amount shown in this table.

The actual number of pages written in a time interval can be obtained from the OMEGAMON PE statistics report. For an example, see **E** in Example 7-2 on page 355.

*Table 7-6 Number of pages that DB2 can write in a single I/O operation*

Page Size	Number of Pages
4 KB	32
8 KB	16
16 KB	8
32 KB	4

## **7.3.6 Tuning buffer pool write frequency**

Large buffer pools benefit DB2 by keeping data pages longer in storage, thus avoiding an I/O operation. With large buffer pools and high write thresholds, DB2 can write large amounts of data at system checkpoint time, and affect performance.

### **Tuning buffer pools using online commands**

The DISPLAY BUFFERPOOL and ALTER BUFFERPOOL commands allow you to monitor and tune buffer pools on line, while DB2 is running, without the overhead of running traces. See *DB2 Version 9.1 for z/OS Command Reference*, SC18-9844 for details about the available DISPLAY BPOOL options.

Figure 7-1 on page 342 shows the output of the DISPLAY BUFFERPOOL command.

*Example 7-1 DIS BUFFERPOOL with DETAIL output*

---

```

-DB9A DIS BUFFERPOOL(BP0) DETAIL
DSNB401I -DB9A BUFFERPOOL NAME BP0, BUFFERPOOL ID 0, USE COUNT 44
DSNB402I -DB9A BUFFER POOL SIZE = 20000 BUFFERS AUTOSIZE = NO 088
          ALLOCATED      =    20000    TO BE DELETED    =        0
          IN-USE/UPDATED  =         8    BUFFERS ACTIVE  =    20000
DSNB406I -DB9A PGFIX ATTRIBUTE - 089
          CURRENT = NO
          PENDING = NO
          PAGE STEALING METHOD = LRU
DSNB404I -DB9A THRESHOLDS - 090
          VP SEQUENTIAL    = 80
          DEFERRED WRITE   = 30    VERTICAL DEFERRED WRT = 5, 0
          PARALLEL SEQUENTIAL =50    ASSISTING PARALLEL SEQT= 0
DSNB409I -DB9A INCREMENTAL STATISTICS SINCE 17:34:48 MAY 13, 2010
DSNB411I -DB9A RANDOM GETPAGE    = 1292 SYNC READ I/O (R) = A 09213

          SEQ. GETPAGE    = 1837856 SYNC READ I/O (S) = B 3
          DMTH HIT        =         0 PAGE-INS REQUIRED =         16
DSNB412I -DB9A SEQUENTIAL PREFETCH - 093
          REQUESTS D =    57472    PREFETCH I/O C =    57471
          PAGES READ E = 1839043
DSNB413I -DB9A LIST PREFETCH - 094
          REQUESTS        =         0    PREFETCH I/O    =         0
          PAGES READ      =         0
DSNB414I -DB9A DYNAMIC PREFETCH - 095
          REQUESTS        =         0    PREFETCH I/O    =         0
          PAGES READ      =         0
DSNB415I -DB9A PREFETCH DISABLED - 096
          NO BUFFER       =         0    NO READ ENGINE  =         0
DSNB420I -DB9A SYS PAGE UPDATES F = 22 SYS PAGES WRITTEN = G 09711

          ASYNC WRITE I/O =         3    SYNC WRITE I/O  =         1
          PAGE-INS REQUIRED =         0
DSNB421I -DB9A DWT HIT          =         0    VERTICAL DWT HIT = 098
DSNB440I -DB9A PARALLEL ACTIVITY - 099
          PARALLEL REQUEST =         0    DEGRADED PARALLEL=         0

DSNB441I -DB9A LPL ACTIVITY - 100
          PAGES ADDED      =         0
DSNB9022I -DB9A DSNB1CMD '-DIS BUFFERPOOL' NORMAL COMPLETION

```

---

In Example 7-1, find the following fields that are highlighted and described as follows:

- A SYNC READ I/O (R) where A shows the number of random synchronous read I/O operations.
- B SYNC READ I/O (S) where B shows the number of sequential synchronous read I/O operations. Sequential synchronous read I/Os occur when prefetch is disabled.  
To determine the total number of synchronous read I/Os, add SYNC READ I/O (S) and SYNC READ I/O (R). In our example, this total is 9216.
- C PREFETCH I/O where C shows the number of times that sequential prefetch occurred.



D In message DSNB412I, REQUESTS where D shows the number of times that sequential prefetch was triggered.

E PAGES READ where E shows the number of pages read using sequential prefetch.

F SYS PAGE UPDATES where F corresponds to the number of buffer updates

G SYS PAGE WRITTEN where G corresponds to the number of buffer written out.

DB2 administrators can tune virtual buffer pool parameters to cause more frequent writes to disk and reduce the impact of the writes at system checkpoint. The tuning parameters are the DWQT and the VDWQT. The DWQT works at virtual buffer pool level, while the VDWQT works at data set level.

Table spaces containing pages that are frequently reread and updated should have a high threshold, placing them in a virtual buffer pool with a high DWQT, or high VDWQT. This way ensures that pages are reused in storage. Large table spaces, where updates are scattered and page reuse is infrequent or improbable, can have their threshold set low, even to zero. A zero threshold means that updated pages are written to disk frequently. In this case, the probability of finding the update page still on the disk cache is higher (cache hit) helping with disk performance. A low threshold also reduces the write impact at checkpoint time.

You might improve the performance of I/O operations by increasing the size of your buffer pools. Make buffer pools as large as you can afford for the following reasons:

- ▶ Using larger buffer pools might mean fewer I/O operations and therefore faster access to your data.
- ▶ Using larger buffer pools can reduce I/O contention for the most frequently used tables and indexes.
- ▶ Using larger buffer pools can speed sorting by reducing I/O contention for work files.

However, many factors affect how you determine the number of buffer pools to have and how big they should be.

Example 7-2 shows a batch report using OMEGAMON for DB2 on z/OS Performance Expert.

*Example 7-2 OMEGAMON PE buffer pool statistics trace report*

BPO	WRITE OPERATIONS	QUANTITY
-----	-----	-----
BUFFER UPDATES		60.00
PAGES WRITTEN		40.00
BUFF.UPDATES/PAGES WRITTEN <b>C</b>		1.50
SYNCHRONOUS WRITES <b>D</b>		12.00
ASYNCHRONOUS WRITES		12.00
PAGES WRITTEN PER WRITE I/O <b>E</b>		1.67
HORIZ.DEF.WRITE THRESHOLD		0.00
VERTI.DEF.WRITE THRESHOLD		0.00
DM THRESHOLD <b>F</b>		0.00
WRITE ENGINE NOT AVAILABLE <b>G</b>		0.00
PAGE-INS REQUIRED FOR WRITE <b>H</b>		0.00

## The buffer pool hit ratio

*Buffer pool hit ratio* is a measure of how often a page access (a getpage operation) is satisfied without requiring an I/O operation. You can help your applications and queries by making the buffer pools large enough to increase the buffer hit ratio.

Accounting reports, which are application-related, show the hit ratio for specific applications. An accounting trace report shows the ratio for single threads. The OMEGAMON buffer pool statistics report shows the hit ratio for the subsystem as a whole. For example, the buffer-pool hit ratio is shown in field **A** in Example 7-3.

*Example 7-3 OMEGAMON PE database buffer pool statistics*

BPO	READ OPERATIONS	QUANTITY	/SECOND	/THREAD	/COMMIT
-----	-----	-----	-----	-----	-----
BPOOL	HIT RATIO (%) <b>A</b>	0.57			
GETPAGE REQUEST		87790.00	12.61	17.6K	609.65
GETPAGE REQUEST-SEQUENTIAL		87023.00	12.50	17.4K	604.33
GETPAGE REQUEST-RANDOM		767.00	0.11	153.40	5.33
SYNCHRONOUS READS <b>B</b>		4.00	0.00	0.80	0.03
SYNCHRON. READS-SEQUENTIAL		4.00	0.00	0.80	0.03
SYNCHRON. READS-RANDOM		0.00	0.00	0.00	0.00
GETPAGE PER SYN.READ-RANDOM		N/C			
SEQUENTIAL PREFETCH REQUEST		2728.00	0.39	545.60	18.94
SEQUENTIAL PREFETCH READS		2728.00	0.39	545.60	18.94
PAGES READ VIA SEQ.PREFETCH <b>C</b>		87282.00	12.54	17.5K	606.13
S.PRF.PAGES READ/S.PRF.READ		31.99			
LIST PREFETCH REQUESTS		0.00	0.00	0.00	0.00
LIST PREFETCH READS		0.00	0.00	0.00	0.00
PAGES READ VIA LIST PREFETCH <b>D</b>		0.00	0.00	0.00	0.00
L.PRF.PAGES READ/L.PRF.READ		N/C			
DYNAMIC PREFETCH REQUESTED		0.00	0.00	0.00	0.00
DYNAMIC PREFETCH READS		0.00	0.00	0.00	0.00
PAGES READ VIA DYN.PREFETCH <b>E</b>		0.00	0.00	0.00	0.00
D.PRF.PAGES READ/D.PRF.READ		N/C			
PREF.DISABLED-NO BUFFER		0.00	0.00	0.00	0.00
PREF.DISABLED-NO READ ENG		0.00	0.00	0.00	0.00
PAGE-INS REQUIRED FOR READ		11.00	0.00	0.58	0.04

The buffer hit ratio uses the following formula to determine how many getpage operations did not require an I/O operation:

$$\text{Hit ratio} = (\text{getpages} - \text{pages\_read\_from\_disk}) / \text{getpages}$$

In the formula, *pages\_read\_from\_disk* is the sum of the following fields:

- ▶ Number of synchronous reads (**B**)
- ▶ Number of pages read via sequential prefetch (**C**)
- ▶ Number of pages read via list prefetch (**D**)
- ▶ Number of pages read via dynamic prefetch (**E**)

If you have 1000 getpages and 100 pages were read from disk, the equation would be as follows:

Hit ratio =  $(1000-100)/1000$

In this case, 0.9 is the hit ratio.

## **Highest and lowest hit ratios**

We define the highest and lowest ratios and other factors.

### ***Highest hit ratio***

The highest possible value for the hit ratio is 1.0, which is achieved when every page requested is always in the buffer pool. Reading index non-leaf pages tends to have a very high hit ratio because they are frequently re-referenced, and therefore, tend to stay in the buffer pool.

### ***Lowest hit ratio***

The lowest hit ratio occurs when the requested page is not in the buffer pool; in this case, the hit ratio is 0 or less. A negative hit ratio means that prefetch has brought pages into the buffer pool that are not subsequently referenced. The pages are not referenced because either the query stops before it reaches the end of the table space or DB2 must take the pages away to make room for newer ones before the query can access them, which can happen when the buffer pool is too small or too few buffer pools are used for many and separate applications.

### ***A low hit ratio is not always bad***

Although making the buffer hit ratio as close to 1.0 as possible might seem desirable, do not automatically assume that a low buffer-pool hit ratio is bad. The hit ratio is a relative value, based on the type of application. For example, an application that browses huge amounts of data using table space scans might well have a buffer-pool hit ratio of 0. What you want to watch for is those cases where the hit ratio drops significantly for the same application. In those cases, investigating further might be helpful.

### ***Checking for wait times***

You can also check OTHER READ I/O WAIT in accounting class 3 to check whether requested pages are read-in without any wait time or wait times occurred.

### ***Hit ratios for additional processes***

The hit ratio measurement becomes less meaningful if the buffer pool is being used by additional processes, such as work files or utilities. Certain utilities and SQL statements use a special type of getpage request that reserves an empty buffer without requiring the page to be read from disk.

A getpage is issued for each empty work file page without read I/O during sort input processing. The hit ratio can be calculated if the work files are isolated in their own buffer pools. If they are, the number of getpages used for the hit ratio formula is divided in half as follows:

Hit ratio =  $((\text{getpages} / 2) - \text{pages\_read\_from\_disk}) / (\text{getpages} / 2)$

## 7.4 Log writes

Log records are created by application programs when data is updated. Each data update requires two log records, one with the data before the update, and another with the data after the update, generally combined into one physical record.

The DB2 log consists of VSAM data sets that are used to register data changes and other significant DB2 events as they occur.

DB2 uses these logs to recover database objects, rollback transactions, perform restart recovery, and record (store) certain diagnostic data. Several major log record types are as follows:

- ▶ System checkpoint records
- ▶ Unit of Recovery (UR) and control records:
  - UR records: Begin UR, end commit phases 1 and 2
  - System checkpoint records
  - Commit database
  - UNDO / REDO records
  - Database exception table (DBET) records
  - Exception states such as LPL, RECP, STOPPED and others

In addition, the DB2 logs also contain SYSCOPY records for SYSUTILX, DBD01, SYSCOPY, page set open and close records, and page set write (diagnostic) records.

### 7.4.1 Log sizing parameters

In the DSNTIPL installation panel (Figure 7-3), you specify the OUTPUT BUFFER, CHECKPOINT FREQ, and FREQUENCY TYPE.

DSNTIPL                  INSTALL DB2 - ACTIVE LOG DATA SET PARAMETERS		
====>		
Enter data below:		
1 <b>NUMBER OF LOGS</b>	====> 3	Data sets per active log copy (2-93)
2 <b>OUTPUT BUFFER</b>	====> 4000K	Size in bytes (40K-400000K)
3 <b>ARCHIVE LOG FREQ</b>	====> 24	Hours per archive run
4 <b>UPDATE RATE</b>	====> 3600	Updates, inserts, and deletes per hour
5 <b>LOG APPLY STORAGE</b>	====> 100M	Maximum ssnmDBM1 storage in MB for fast log apply (0-100M)
6 <b>CHECKPOINT FREQ</b>	====> 500000	Log records or minutes per checkpoint
7 <b>FREQUENCY TYPE</b>	====> LOGRECS	CHECKPOINT FREQ units. LOGRECS, MINUTES
8 <b>UR CHECK FREQ</b>	====> 0	Checkpoints to enable UR check. 0-255
9 <b>UR LOG WRITE CHECK</b>	====> OK	Log Writes to enable UR check. 0-1000K
10 <b>LIMIT BACKOUT</b>	====> AUTO	Limit backout processing. AUTO, YES, NO
11 <b>BACKOUT DURATION</b>	====> 5	Checkpoints processed during backout if LIMIT BACKOUT = AUTO or YES. 0-255
12 <b>RO SWITCH CHKPTS</b>	====> 5	Checkpoints to read-only switch.
1-32767		
13 <b>RO SWITCH TIME</b>	====> 10	Minutes to read-only switch. 1-32767
14 <b>LEVELID UPDATE FREQ</b>	====> 5	Checkpoints between updates. 0-32767

Figure 7-3 Install Panel for log data sets

The capacity that you specify for the active log affects DB2 performance significantly. If you specify a capacity that is too small, DB2 might need to access data in the archive log during rollback, restart, and recovery processes. Accessing an archive takes a considerable amount of time especially if the archive log is on tape.

The following DB2 parameters affect the capacity of the active log. In each case, increasing the value the system administrator specifies for the parameter increases the capacity of the active log. See Section 2 of the *DB2 Version 9.1 for z/OS Installation Guide*, GC18-9846, for more information about updating the active log parameters.

The main parameters on the installation panel DSNTIPL (Figure 7-3 on page 358) are as follows:

- ▶ NUMBER OF LOGS controls the number of active log data sets.
- ▶ OUTPUT BUFFER specifies the size of the output buffer used for writing active log data sets.
- ▶ ARCHIVE LOG FREQ controls how often active log data sets are copied to the archive log.
- ▶ UPDATE RATE is an estimate of how many database changes (inserts, update, and deletes) are expected per hour.
- ▶ CHECKPOINT FREQ specifies the number of log records that DB2 writes between checkpoints.

The DB2 installation CLIST uses ARCHIVE LOG FREQ and UPDATE RATE parameters to calculate the data set size of each active log data set.

## 7.4.2 Improving log write performance

To improve log-write performance, consider using any of the following approaches:

- ▶ If you replicate your logs to remote sites, choose the storage system that provide the best possible performance for remote replication.
- ▶ Choose the correct size that your system can tolerate for the log output buffer. Because the pages for the log output buffer becomes fixed in real storage, be sure you are not causing issues with real storage. A larger size for the log output buffer might help only for ROLLBACK situations. A smaller size can increase the number of forced I/O operations that occur because additional buffers are unavailable, and can also increase the number of wait conditions.

You can use the OUTPUT BUFFER field (the OUTBUFF subsystem parameter) of installation panel DSNTIPL to specify the size of the output buffer used for writing active log data sets. The maximum size of the log output buffer is 400,000 KB. To validate the OUTBUFF setting, you can collect IFCID 0001 (system services statistics) trace records. The QJSTWTB field indicates the number of times the buffer was full and caused a log record to wait for I/O to complete. A non-zero count for QJSTWTB might indicate that the log output buffer is too small.

- ▶ Choose fast devices for log data sets. The devices that are assigned to the active log data sets must be fast. In environments with high levels of write activity, use high-capacity storage systems, such as the IBM TotalStorage DS8000 series, to avoid logging bottlenecks.
- ▶ Avoid device contention. Place the copy of the bootstrap data set and, if using dual active logging, the copy of the active log data sets, on volumes that are accessible on a path different than that of their primary counterparts. Preformat new active log data sets. Whenever you allocate new active log data sets, preformat them using the DSNJLOGF

utility. This action avoids the overhead of preformatting the log, which normally occurs at unpredictable times.

- ▶ Stripe the active log data sets. The active logs can be striped using DFSMS. Striping is a technique to improve the performance of data sets that are processed sequentially. Striping is achieved by splitting the data set into segments or stripes and spreading those stripes across multiple volumes. Striping can improve the maximum throughput log capacity and is most effective when many changes to log records occur between commits. Striping is useful if you have a high I/O rate for the logs. Striping is needed more with ESCON channels than with the faster FICON channels.
- ▶ Stripe the archive log data sets on disk. If write operations to the archive do not complete as fast as write operations to the active log, transactions might slow down while waiting for the active log to be emptied. When applying log records from the archive log, striping helps DB2 read the archive data sets faster.

### **Two-phase commit log-writes**

Because they use two-phase commit, applications that use the CICS, IMS, and RRS attachment facilities might force writes to the log twice:

- ▶ The first write forces all the log records of changes to be written (if they have not been written previously because of the write threshold being reached).
- ▶ The second write writes a log record that takes the unit of recovery into an in-commit state.

### **Physical writes**

Figure 7-4 on page 361 also shows the physical writes to disk. The log records in the log output buffer are written from the output buffer to disk.

DB2 uses two types of log writes, asynchronous and synchronous, which are explained in 7.4.3, “Asynchronous writes” on page 361 and 7.3.2, “Synchronous writes” on page 351.

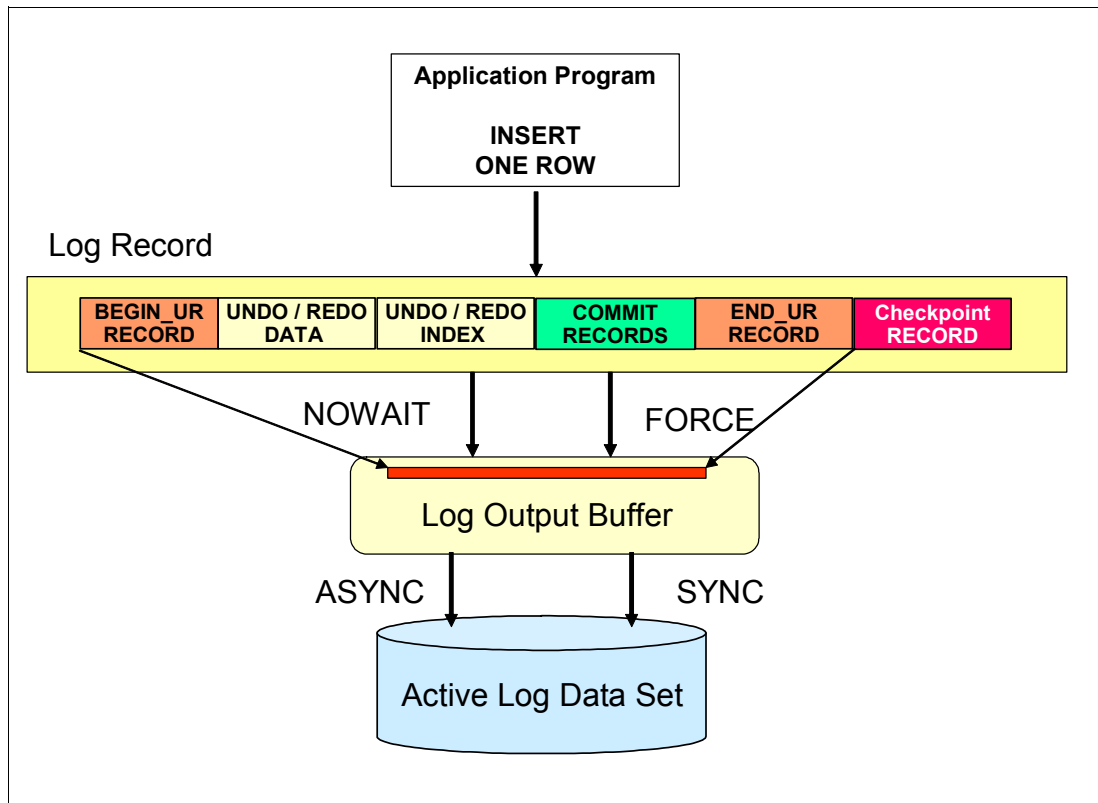


Figure 7-4 Log record path to disk

### 7.4.3 Asynchronous writes

Asynchronous writes are the most common. These asynchronous writes occur when data is updated. The before and after image records are usually moved to the log output buffer, and control is returned to the application. However, if no log buffer is available, the application must wait for one to become available.

### 7.4.4 Synchronous writes

Synchronous writes usually occur at commit-time when an application has updated data. This write is called *forcing* the log, because the application must wait for DB2 to write the log buffers to disk before control is returned to the application. If the log data set is not busy, all log buffers are written to disk. If the log data set is busy, the requests are queued until it is freed.

### 7.4.5 Writing to two logs

If two logs exist (recommended for availability), the writing to the first log, in general, must complete before the writing to the second log begins. The first time a log control interval is written to disk, the write I/Os to the log data sets are done in parallel. However, if the same 4 KB log control interval is again written to disk, then the write I/Os to the log data sets must be done serially to prevent any possibility of losing log data in case I/O errors occur simultaneously on both copies. This method improves system integrity. I/O overlap in dual logging occurs when multiple log control intervals have to be written, for example, when the

WRITE THRESHOLD value is reached, or when log records accumulate because of a log-device-busy condition.

## 7.5 Log reads

During a rollback, restart, and database recovery is when the performance effect of log-reads becomes evident. DB2 must read from the log and apply changes to the data on disk. Every process that requests a log-read has an input buffer dedicated to that process. DB2 optimizes the log reads searching for log records in the following order:

1. Log output buffer
2. Active log data set
3. Archive log data set

If the log records are in the output buffer, DB2 reads the records directly from that buffer. If the log records are in the active or archive log, DB2 moves those log records into the input buffer used by the reading process (such as a recovery job or a rollback).

Access is reported by OMEGAMON PE; see Example 7-4 on page 364. From a performance perspective, it is always best for DB2 to obtain the log records from the output buffer. The next fastest access for DB2 is the active log. Access to the archive log is not desirable; it can be delayed for a considerable length of time. For example, tape drives might not be available, or a tape mount can be required.

### 7.5.1 Improving log-read performance

In this section, we present choices to improve log-read performance.

#### **Active log size**

Active logs must be large enough to avoid reading the archives, especially during restart, rollback, and recovery processes. When data is backed out, performance is optimal if the data is available from the output buffer or from the active log. If the data is no longer available from the active log, the active log is probably too small. For information about sizing the active log data sets, see 7.5.2, “Active log size” on page 363.

#### **Avoid device contention**

Avoid device contention on the log data sets. See guidelines in 7.4.2, “Improving log write performance” on page 359.

#### **Archive to disk or tape**

If the archive log data set resides on disk, it can be shared by many log readers. In contrast, an archive on tape cannot be shared among log readers. Although a good practice is to avoid reading archives altogether, if a process must read the archive, that process is serialized with anyone else who must read the archive tape volume. For example, every rollback that accesses the archive log must wait for any previous rollback work that accesses the same archive tape volume to complete and deadlocks can happen in data sharing.

If the log records are in the output buffer, DB2 reads the records directly from that buffer. If the log records are in the active or archive log, DB2 moves those log records into the input buffer that is used by the reading process (such as a recovery job or a rollback).



DB2 reads the log records faster from the active log than from the archive log. Access to archived information can be delayed for a considerable length of time if a unit is unavailable or if a volume mount is required (for example, a tape mount).

To improve log read performance, consider implementing the following options:

- ▶ **Archive to disk.** If the archive log data set resides on disk, it can be shared by many log readers. In contrast, an archive on tape cannot be shared among log readers. As previously mentioned, a good practice is to avoid reading archives altogether, however if a process must read the archive, that process is serialized with anyone else who must read the archive tape volume. If you do not have enough space to maintain the archive data sets on disk, consider using DFHSM to write the archive data sets to tape. This method has a disadvantage in that HSM must read the archive data set from disk to write it to tape, but the recall process is improved for a number of reasons. You can pre-stage the recalls of archive data sets in parallel (to striped data sets); when the data sets are recalled, parallel readers can proceed.

Archiving to disk offers several advantages:

- Recovery times can be reduced by eliminating tape mounts and rewind time for archive logs kept on tape.
  - Multiple *recover* utilities can be run in parallel.
  - DB2 log data can span a greater length of time than what is currently kept in your active log data sets.
  - The need for tape drives during DB2 archive log creation is eliminated. If DB2 needs to obtain a tape drive on which to create the archive logs and it cannot allocate one, all activity eventually stops unless DB2 can create the archive log data sets.
- ▶ **Avoid device contention on the log data sets** by placing your active log data sets on different volumes and I/O paths to avoid I/O contention in periods of high concurrent log read activity. When multiple concurrent readers access the active log, DB2 can ease contention by assigning some readers to a second copy of the log. Therefore, for performance and error recovery, use dual logging and place the active log data sets on a number of separate volumes and I/O paths. When possible, put data sets within a copy or within separate copies on separate volumes and I/O paths. Ensure that no data sets for the first copy of the log are on the same volume as data sets for the second copy of the log.
  - ▶ **Stripe active log data sets.** The active logs can be striped by using DFSMS. Striping is a technique to improve the performance of data sets that are processed sequentially. Striping is achieved by splitting the data set into segments or stripes and spreading those stripes across multiple volumes. Striping can improve the maximum throughput log capacity and is most effective when many changes to log records occur between commit operations. Striping is useful if you have a high I/O rate for the logs. Striping is needed more with ESCON channels than with the faster FICON channels.

## 7.5.2 Active log size

The capacity the system administrator specifies for the active log can affect DB2 performance significantly. If the capacity is too small, DB2 might have to access data in the archive log during rollback, restart, and recovery. Accessing an archive log generally takes longer than accessing an active log.

### Gathering statistics

You can gather statistics for logging activities from the OMEGAMON statistics report.

A non-zero value for **A** in Example 7-4 indicates that your output buffer is too small. Ensure that the size you choose is backed up by real storage. A non-zero value for **B** is an indicator that your output buffer is too large for the amount of available real storage.

*Example 7-4 OMEGAMON statistics report: log activity*

LOG ACTIVITY	QUANTITY	/SECOND	/THREAD	/COMMIT
READS SATISFIED-OUTPUT BUFF	0.00	0.00	0.00	0.00
READS SATISFIED-OUTP.BUF(%)	N/C			
READS SATISFIED-ACTIVE LOG	0.00	0.00	0.00	0.00
READS SATISFIED-ACTV.LOG(%)	N/C			
READS SATISFIED-ARCHIVE LOG	0.00	0.00	0.00	0.00
READS SATISFIED-ARCH.LOG(%)	N/C			
TAPE VOLUME CONTENTION WAIT	0.00	0.00	0.00	0.00
READ DELAYED-UNAVAIL.RESOUR	0.00	0.00	0.00	0.00
ARCHIVE LOG READ ALLOCATION	0.00	0.00	0.00	0.00
ARCHIVE LOG WRITE ALLOCAT.	0.00	0.00	0.00	0.00
CONTR.INTERV.OFFLOADED-ARCH	0.00	0.00	0.00	0.00
LOOK-AHEAD MOUNT ATTEMPTED	0.00	0.00	0.00	0.00
LOOK-AHEAD MOUNT SUCCESSFUL	0.00	0.00	0.00	0.00
<b>UNAVAILABLE OUTPUT LOG BUFF</b>	<b><u>A</u> 0.00</b>	0.00	0.00	0.00
<b>OUTPUT LOG BUFFER PAGED IN</b>	<b><u>B</u> 10.00</b>	0.00	5.00	0.15
<b>LOG RECORDS CREATED</b>	<b><u>C</u> 112.00</b>	0.03	56.00	1.65
<b>LOG CI CREATED</b>	<b><u>D</u> 35.00</b>	0.01	17.50	0.51
LOG WRITE I/O REQ (LOG1&2)	66.00	0.02	33.00	0.97
LOG CI WRITTEN (LOG1&2)	114.00	0.03	57.00	1.68
LOG RATE FOR 1 LOG (MB)	N/A	0.00	N/A	N/A
LOG WRITE SUSPENDED	22.00	0.01	11.00	0.32

## Calculating average log record size

One way to determine how much log volume is needed is to calculate the average size, in bytes, of log records that are written.

As a general estimate, start with 200 bytes as the average size. To increase the accuracy of this estimate, get the real size of the log records that are written.

As an example, to calculate the average size of log records that are written, obtain the statistics from Example 7-4:

1. Collect the following values from the statistics report:
  - LOG RECORDS CREATED  
This value is the number of log records created (**C**)
  - LOG CI CREATED  
This value is the number of control intervals created in the active log counter (**D**)
2. Use the following formula:  
Average size of log record in bytes = **D** \* 4096 / **C**

Using this value to estimate logging needs, plus considering the available device sizes, the DB2 system administrator can update the output of the installation CLIST to modify the calculated values for active log data set sizes.

## 7.6 Distributing data sets more efficiently

Avoid I/O contention and increase throughput through the I/O subsystem by placing frequently used data sets on fast disk devices and by distributing I/O activity.

However, distributing I/O activity is less important when you use disk devices with parallel access volumes (PAV) support and multiple allegiance support.

### Putting frequently used data sets on fast devices

You can improve performance by assigning frequently used data sets to faster disk devices.

To make the best use of your disk devices, consider the following guidelines:

- ▶ Assign the most frequently used data sets to the faster disk devices at your disposal.
- ▶ For partitioned table spaces, consider having some partitions on faster devices. Placing frequently used data sets on fast disk devices also improves performance for nonpartitioned table spaces.
- ▶ Partition any nonpartitioned table spaces that have excessive I/O contention at the data set level.

See Chapter 4, “System managed DB2 data” on page 193.

### Distributing the I/O

By distributing your data sets, you can prevent I/O requests from being queued in z/OS.

To distribute I/O operations, consider the following guidelines:

- ▶ If you do not have parallel access volumes, allocate frequently used data sets or partitions across your available disk volumes so that I/O operations are distributed. Even with RAID devices, in which the data set is spread across the physical disks in an array, data should be accessed at the same time on separate logical volumes to reduce the chance of an I/O request being queued in z/OS.
- ▶ Isolate data sets that have characteristics that do not complement other data sets.

### Using partitioning schemes and data clustering for partitioned table spaces

Depending on the type of operations that your applications emphasize, you have several options for distributing I/O.

If the partitions of your partitioned table spaces must be of relatively the same size (which can be a great benefit for query parallelism), consider using a ROWID column as all or part of the partitioning key.

For partitions that are of such unequal size that performance is negatively affected, alter the limit key values to set new partition boundaries and then reorganize the affected partitions to rebalance the data. Alternatively, you can use the REORG utility with the REBALANCE keyword to set new partition boundaries.

REBALANCE causes DB2 to change the limit key values such that the rows in the range of partitions being reorganized are distributed across those partitions as evenly as possible.

If your performance objectives emphasize inserts, distribute the data in a manner that reduces the amount of clustered key values. Consider designing your database with randomized index keys design to remove clustering. You can also take advantage of the index

page splitting by choosing the appropriate size for index pages. If the rate of inserts does not require you to spread out the inserts, consider creating or altering tables with the APPEND YES option.

In contrast, for performance objectives that emphasize read operations, your data clustering should reflect the sequence in which queries will be processed so that DB2 can use the sequential processing method of parallelism to reduce I/O and CPU time.

Partition data that will be subject to frequent update operations in a manner that provides plenty of free space, especially if new and updated rows might expand because they contain columns with varying-length data types or compressed data

### 7.6.1 Creating additional work file table spaces to reduce contention

You can minimize I/O contention in certain situations by creating additional work file table spaces.

For a single query, the recommended number of work file disk volumes to have is one-fifth the maximum number of data partitions, with 5 - 50 (minimum to maximum). For concurrently running queries, multiply this value by the number of concurrent queries. Depending on the number of table spaces and the amount of concurrent activity, performance can vary. In general, adding more table spaces improves performance.

In query parallelism environments, ensure that the number of work file disk volumes is at least equal to the maximum number of parallel operation use for queries in a given workload, place these volumes on separate channel or control unit paths, and monitor the I/O activity for the work file table spaces.

1. Place the volumes on separate channel or control unit paths.
2. Monitor the I/O activity for the work file table spaces. You might need to further separate this work file activity to avoid contention.
3. As the amount of work file activity increases, consider increasing the size of the buffer pool for work files to support concurrent activities more efficiently. The general guideline for the work file buffer pool is to increase the size to obtain the following buffer pool statistics:
  - MERGE PASSES DEGRADED must be less than 1% of MERGE PASS REQUESTED.
  - WORKFILE REQUESTS REJECTED must be less than 1% of WORKFILE REQUEST ALL MERGE PASSES.
  - Synchronous read I/O must be less than 1% of pages read by prefetch.
  - Prefetch quantity should be near eight.
4. If your applications require extensive use of temporary objects or operations that can be processed in work files that span more than one table space, define multiple table spaces in the work file database that have a zero secondary quantity and are stored in DB2-managed data sets. Processing that uses work files and can span more than one table space includes objects and operations such as those in the following list:
  - Large concurrent sorts and single large sorts
  - Created temporary tables
  - Several merge, star, and outer joins
  - Non-correlated subqueries
  - Materialized views
  - Materialized nested table expressions
  - Triggers with transition variables

When a table space in the work file database is stored in user-managed data sets, DB2 does not detect during table space selection whether any secondary allocation exists. Consequently when space is allocated for work files, such table spaces are given the same preference as table spaces that are stored in DB2-managed data sets and have a zero secondary quantity, even when the table space has a secondary allocation.

DB2 gives preference to table spaces that are stored in DB2-managed data sets and have zero secondary quantity when allocating space for work files for processing that can span more than one table space. By creating multiple work file table spaces with zero secondary quantity, you support efficient concurrent read and write I/Os to the work files.

5. If your applications require extensive use of temporary objects or operations that can be processed only in a single table space, define some table spaces in the work file database that have a non-zero secondary quantity and are stored in DB2-managed data sets. Processing that uses work files and is limited to a single table space includes the following objects and operations, to name several:
  - Declared global temporary tables
  - Scrollable cursors
  - SQL MERGE statements

The table spaces with non-zero secondary quantity help to minimize contention for space between temporary objects or operations that can span multiple table spaces and those that cannot. DB2 gives preference for processing that cannot span more than one table space to table spaces that are stored in DB2-managed data sets and that can grow beyond the primary space allocation because the secondary quantity is greater than zero.

6. Ensure that the work file database contains at least one 32-KB page size table space before you create declared global temporary tables. The rows of declared global temporary tables reside in a table space in the work file database, and DB2 does not create an implicit table space for the declared global temporary table.
7. To further improve performance, consider allocating more 32-KB data sets. DB2 uses 32 KB work file data sets when the total work file record size, including record overhead, exceeds 100 bytes, resulting in better performance and reduced work file buffer pool and disk space usage for work files, in some cases.

## 7.6.2 Formatting early and speeding-up formatting

You can improve the performance of applications that use heavy insert processing by allocating space so that cylinders are used as the allocation amount, and by pre-formatting a table space before inserting data.

### Allocating space in cylinders or in large primary and secondary quantities

Specify your space allocation amounts to ensure allocation by cylinder. If you use record allocation for more than a cylinder, cylinder allocation is used. Cylinder allocation can reduce the time required to do SQL mass inserts and to perform LOGONLY recovery; it does not affect the time required to recover a table space from an image copy or to run the REBUILD utility.

When inserting records, DB2 pre-formats space within a page set as needed. The allocation amount, which is either CYLINDER or TRACK, determines the amount of space that is pre-formatted at any one time.

Because less space is pre-formatted at one time for the TRACK allocation amount, a mass insert can take longer when the allocation amount is TRACK than the same insert when the

allocation amount is CYLINDER. However, smart secondary space allocation minimizes the difference between TRACK and CYLINDER.

The allocation amount is dependent on device type and the number of bytes you specify for PRIQTY and SECQTY when you define table spaces and indexes. The default SECQTY value is 10% of the PRIQTY value, or three times the page size, whichever is larger. This default quantity is an efficient use of storage allocation. Choosing a SECQTY value that is too small in relation to the PRIQTY value results in track allocation

### Avoiding excessively small extents

Data set extent size affects performance because excessively small extents can degrade performance during a sequential database scan.

As an example, suppose that the sequential data transfer speed is 100 MBps and that the extent size is 10 MB. The sequential scan must move to a new extent ten times per second.

**Note:** Consider maintaining extent sizes that are large enough to avoid excessively frequent extent moving during scans. Keep the extent size greater than 10 cylinders for large data sets.

### Maximum number of extents

An SMS-managed linear data set is limited to 123 extents on a volume and 7257 total extents on all volumes. A non-SMS-managed data set is limited to 123 extents on a volume and 251 total extents on all volumes. If a data set grows and extents are not monitored, jobs eventually fail because of these extent limitations.

**Note:** Monitor the number of extents to avoid reaching the maximum number of extents on a volume and the maximum number of extents on all volumes.

### Specifying primary quantity for non-partitioned indexes

Specifying sufficient primary and secondary allocations for frequently used data sets minimizes I/O time, because the data is not located at different places on the disks.

Listing the catalog or VTOC occasionally to determine the number of secondary allocations that have been made for your more frequently used data sets can also be helpful.

Alternatively, you can use IFCID 0258 in the statistics class 3 trace and real-time statistics to monitor data set extensions. OMEGAMON monitors IFCID 0258.

To prevent wasted space for non-partitioned indexes, take one of the following actions:

- ▶ Let DB2 use the default primary quantity and calculate the secondary quantities. Do this by specifying 0 for the IXQTY subsystem parameter, and by omitting a PRIQTY and SECQTY value in the CREATE INDEX statement or ALTER INDEX statement. If a primary and secondary quantity were previously specified for an index, you can specify PRIQTY -1 and SECQTY -1 to change to the default primary quantity and calculated secondary quantity.
- ▶ If the MGEXTSZ subsystem parameter is set to NO, so that you control secondary space allocations, make sure that the value of  $\text{PRIQTY} + (\text{N} * \text{SECQTY})$  is a value that evenly divides into PIECESIZE.

## 7.7 DB2 sort work files

The work files that are used in sort are logical work files, which reside in work file table spaces in your work file database (which is DSNDB07 in a non-data-sharing environment).

The sort begins with the input phase, when ordered sets of rows are written to work files. At the end of the input phase, when all the rows have been sorted and inserted into the work files, the work files are merged together, if necessary, into a single work file that contains the sorted data. The merge phase is skipped if only one work file exists at the end of the input phase. In some cases, intermediate merging might be needed if the maximum number of sort work files has been allocated.

DB2 uses the buffer pool when writing to the logical work file. Only the buffer pool size limits the number of work files that can be used for sorting.

A sort can complete in the buffer pool without I/Os. This ideal situation might be unlikely, especially if the amount of data being sorted is large. The sort row size is actually made up of the columns being sorted (the sort key length) and the columns that the user selects (the sort data length). Having a very large buffer pool for sort activity can help you to avoid disk I/Os.

When your application needs to sort data, DB2 tries to allocate each sort work file on a table space that has no secondary allocation (SECQTY = 0) and is least recently used. When table spaces with the preferred features are not available, then any available table space will be used.

For example, if five logical work files are to be used in the sort, and the installation has three work file table spaces allocated, then the following table shows which work file table space would contain each logical work file.

For any SQL statement that initiates sort activity, the OMEGAMON SQL activity reports provide information about the efficiency of the sort that is involved.

To minimize the performance impacts of sort processing (and as it relates to disk storage), consider the following guidelines:

- ▶ Increase the size of the sort pool. The larger the sort pool, the more efficient the sort is.
- ▶ Minimize I/O contention on the I/O paths to the physical work files, and make sure that physical work files are allocated on different I/O paths and packs to minimize I/O contention. Using disk devices with PAV support is another way to significantly minimize I/O contention. When I/Os occur in the merge phase of a sort, DB2 uses sequential prefetch to bring pages into the buffer pool with a prefetch quantity of eight pages. However, if the buffer pool is constrained, then DB2 uses a prefetch quantity of four pages or less, or disables prefetch entirely because of the unavailability of enough pages.
- ▶ Allocate additional physical work files in excess of the defaults, and put those work files in their own buffer pool. Segregating work file activity enables you to better monitor and tune sort performance. It also allows DB2 to handle sorts more efficiently because these buffers are available only for sort without interference from other DB2 work.
- ▶ Increase the amount of available space for work files. Applications that use created temporary tables use work file space until a COMMIT or ROLLBACK occurs. (If a cursor is defined WITH HOLD, then the data is held past the COMMIT.) If sorts are happening concurrently with the temporary table's existence, then you probably need more space to handle the additional use of the work files. Applications that require star join, materialized views, materialized nested table expressions, non-correlated subqueries or triggers also use work files.

- ▶ Write applications to sort only columns that need to be sorted because sorted rows appear twice in the sort row size. A smaller sort row size means that more rows can fit in the sort pool.
- ▶ Select VARCHAR columns only when they are required. Varying length columns are padded to their maximum length for sort row size.
- ▶ Set the buffer pool sequential steal threshold (VPSEQT) to 99% unless sparse index is used to access the work files. The default value, which is 80%, allows 20% of the buffers to go unused. A value of 99% prevents space map pages, which are randomly accessed, from being overwritten by massive prefetch.
- ▶ Increase the buffer pool deferred write threshold (DWQT) or data set deferred write threshold (VDWQT) values. If the DWQT or VDWQT are reached, writes are scheduled. For a large sort using many logical work files, this is difficult to avoid, even if a very large buffer pool is specified.

## 7.8 DB2 and tape processing

The role and characteristics of tape storage have changed dramatically over the last 50 years. In the early days of computing, tape was the first electronic media not only to store data on, but it also supported the first operating system. With the fast evolution of disk, tape became the media of choice for large data sets, exchange data, and backup. Cheaper disks with huge capacities became an economical alternative for tape.

Despite all efforts to constrain the use of tape, its use is still growing exponentially. Moreover, high-end tape technology has exploded from 200 MB of raw capacity per cartridge up to today's 500 GB of uncompressed data and even greater capacities are projected during the next couple of years.

### 7.8.1 Virtual tape terminology

In this section, we explain the most common components for tapes.

#### **Automated Tape Library (ATL)**

The ATL is a device consisting of robotic components, cartridge storage areas, tape subsystems, and controlling hardware and software, together with the set of tape volumes that reside in the library and can be mounted on the library tape drives. ATLs do not provide tape virtualization or automatic tape stacking.

#### **Virtual Tape Server (VTS)**

The VTS is a hardware-based solution that addresses not only tape cartridge utilization but also tape device utilization and hence overall tape subsystem costs. The VTS was introduced and made available in 1996 as a new breed of storage solution that combines a high-speed disk cache with tape automation, tape drives, and intelligent storage management software running on a server.

#### **IBM System Storage Virtualization Engine**

The IBM System Storage Virtualization Engine TS7700 is the newest member of the IBM TS7000 Virtualization Family. It represents the fourth generation of IBM Tape Virtualization for mainframe systems and replaces the highly successful IBM TotalStorage Virtual Tape Server.

For more information about TS7700, see *IBM Virtualization Engine TS7700 with R1.6*, SG24-7712.



Storage virtualization emulates IBM 3490 Enhanced Capacity (3490E) tape drives of a specific size: 400, 800, 1000, 2000 or 4000 MB. These emulated tape drives are also called virtual drives. It provides for a large number of virtual devices, up to 256 per TS7740.

From a host perspective, the TS7700 Virtualization Engine is a subsystem that supports up to 16 tape control units, each with 16 IBM 3490E tape drives, for a total of 256 tape drives in a single Cluster configuration.

The emulation is transparent to the host and to applications. The host always writes to and reads from virtual tape drives, it never accesses the physical tape drives in the back end. In fact, it does not need to know these tape drives exist.

The TS7700 keeps its statistical records in a DB2 pSeries® database in the TS7700 controller. This database is not directly accessible to customers.

Initial creation of tape data sets are actually to disk (DS6000), not tape as follows:

- ▶ Process uses a RAID5 disk subsystem (fault tolerant) to store volumes written by the host.
- ▶ Total disk space capacity is 6 TB, and 18 TB when compressed.
- ▶ The disk space provided is called Tape Volume Cache (TVC).
- ▶ All data must be written to or read from emulated volumes in the disk based TVC.
- ▶ Emulated tape volumes in the Tape Volume Cache are called *virtual volumes*.
- ▶ Data that is written from the host into the tape volume cache, is scheduled for being copied to tape at a later point in time

## 7.8.2 Virtual volume size versus physical volume size

For tape allocations, emulation of an IBM 3490 Enhanced Capacity (3490E) tape drive of a specific size: 400, 800, 1000, 2000 or 4000 MB is performed.

Only the amount required is actually allocated to the TVC and tape. For example, we use an SMS Data Class assignment to use 4000 MB, but we only write out 30 MB. In this case, only 30 MB are allocated to the TVC and tape.

Physical tape capacity is much larger, however we can only logically create a volume up to 4 GB. This is why tape stacking is so important.

### Active log data set size when archiving to tape

Many customers size the DB2 active log data sets so that an archive log will fit on one tape. The original sizing was done back in the earlier days when a tape could only hold 400 or 800 MB.

Reevaluate the size of the active log data sets when archiving to tape because of the virtual tape size increased to 4 GB. Active and archive log data sets can be allocated up to 4 GB. For ATLS, the virtual tape size is ignored and physical tape size should be taken into account.

In prior versions of DB2 for z/OS, an active log can have up to 4 GB of tracks on a single disk volume, whereas the size limit for an archive log is only 64000 tracks (about 3 GB). As a result, installations that use 4 GB active logs are forced to use multi-volumes archive or put archive logs on tape. In version 1.7 of z/OS DB2 started to support large data sets and removed the 64 KB archive-log limit. DB2 9 for z/OS takes advantage of the new limit by introducing the DSNTYPE=LARGE attribute of the PRIMARY QUANTITY field of installation panel DSNTIPA. The new attribute supports an archive log that has up to 4 GB of tracks per disk volume.

In the DSNTIPA installation panel of Figure 7-5, if you use the default, a blank, the *install CLIST* calculates this space using block size and size of the log. DFSMS Direct Access Device Space Management (DADSM) limits the space allocation on a single volume to less than 64000 tracks. Therefore, if the archive log data set size can be greater than or equal to 64000 tracks, you must specify a primary space quantity of less than 64000 (2\*\*16) tracks. This forces the archive log data set to extend to a second volume.

DSNTIPA                      INSTALL DB2 - ARCHIVE LOG DATA SET PARAMETERS		
====>		
Enter data below:		
1	ALLOCATION UNITS	====> BLK                      Blk, Trk, or Cyl
2	PRIMARY QUANTITY	====>                      Primary space allocation
3	SECONDARY QTY.	====>                      Secondary space allocation
4	CATALOG DATA	====> NO                      YES or NO to catalog archive data sets
5	DEVICE TYPE 1	====> TAPE                      Unit name for COPY1 archive logs
6	DEVICE TYPE 2	====>                      Unit name for COPY2 archive logs
7	BLOCK SIZE	====> 24576                      Rounded up to 4096 multiple
8	READ TAPE UNITS	====> 2                      Number of allocated read tape units
9	DEALLOC PERIOD	====> 0                      Time interval to deallocate tape units
10	RECORDING MAX	====> 10000                      Number of data sets recorded in BSDS
11	WRITE TO OPER	====> YES                      Issue WTOR before mount for archive
12	WTOR ROUTE CODE	====> 1,3,4
13	RETENTION PERIOD	====> 9999                      Days to retain archive log data sets
14	QUIESCE PERIOD	====> 5                      Maximum quiesce interval (1-999)
15	COMPACT DATA	====> NO                      YES or NO for data compaction
16	SINGLE VOLUME	====> NO                      Single volume for DASD archives. NO or YES

Figure 7-5 DB2 installation panel for ARCHIVE LOG data set parameters

### 7.8.3 DSNZPARMs for archiving to tape

The following parameters affect archiving to tape:

► **ARCWTOR**

Consider setting this parameter to YES when you write the archive logs to a manually mounted tape system and want to get the attention of your operators. Remember, VE and VTS mounts really write to DASD, even though a tape is requested.

► **ARCRETN**

Be realistic about the number of days you want to keep an archive. See MAXARCH parameter for additional information.

► **BLKSIZE**

Use BLKSIZE=24576 if you need flexibility. If you are writing to any type of tape (even VTS and VE) and will never need to copy to disk, use BLKSIZE=28672. For tapes, you dramatically increase overall throughput performance when using large blocks, as compared to 4 KB or 8 KB.

Compression factor if used can decrease throughput for smaller blocks. Consider the following information:

- DFP sees all tape UCBs as real tape and serializes the request, passing that information to DB2. Even when an archive log has been recalled to TVC, which is DISK, DB2 still sees this as a tape object. For archive log parallelism because of mass recoveries, recreate the data set on DASD. The problem here is that a track of DISK can hold 48 KB of data in 4 KB blocks with a block size of 32 KB or less. Had you used

the disk formulated specification of BLKSIZE=24576, you would have had two blocks per track. If you optimize tape and use 28672, then you can create only one block per track and use only about 60% of the track. After an archive data set with a specific block size has been created, it cannot be copied with a new block size.

- Determine how often, realistically, you will re-create an archive data set from tape to disk. Unless it is rare, use BLKSIZE to 24576 instead of 28672. BLKSIZE for tape of 24576 does make a difference, both in terms of throughput performance and cartridge capacity, however it is not that significant.

► PRIQTY, SECQTY, and ALCUNIT

These parameters are ignored during the writing to tape.

► UNIT and UNIT2

Understanding how your company's tape and disk are architected for your data sets is essential. There are a variety of ways to use UNIT and UNIT2, and still have both software dual-copied archive log data sets end up on the same physical volume. The same is true for other software-dual copied data sets, such as image copies. The result is a single point of failure. You can avoid this problem in several ways. One way is if a secondary copy of your software dual copied data sets reside on another tape, preferably at a separate site. Other options are available too, such as using tape Selective Dual Copy, and transmitting a copy of specific data sets to a safe location.

► DEALLCT

If your archive log data sets are created on manual tape, set this value high enough to allow DB2 to optimize tape handling for multiple read applications. When all tape reading is complete, you can update this option with the SET ARCHIVE command. If your archive logs are written to VTS or VE, set it to 0, remember they will be recalled to TVC which is really disk. Do not set this value to 1440 or NOLIMIT unless you follow it with a SET ARCHIVE command, otherwise your tape and the unit will not deallocate until DB2 shuts down.

**Note:** In a data sharing environment, the archive tape is not available to other members of the group until the deallocation period expires. For Data Sharing environments where recoveries are run from multiple members, set DEALLCT=0. If all recoveries are executed from one member, DEALLCT can be a higher value.

► MAXARCH

After you ran the BSDS conversion utility (as was done in DB2 V8), your maximum limit for archive volumes is 10,000 volume slots versus the old value of 1,000. Remember, each slot is a volume, not the entire archive log data set. Assume you are archiving a 4 GB active log, which would require a 4 GB archive log. If you use VTS or VE and you create the data sets on a volume holding 400 MB, you create 10 volumes and therefore use 10 slots. In this case, it would have been more beneficial to write to an SMS Data Class with 4000 MB and create only one volume slot.

**Note:** Assume you set the MAXARCH=1000 and ARCRETN=7. You create 10 tape volumes per archive log data set and have 50 active log full switches average every day. On a daily basis, you use up 500 of the 1,000 slots, meaning you are only holding two days worth of archive in the BSDS. In this example, you are not keeping enough days worth of archive logs in the BSDS to meet your requirements for the number of days you want to keep your archive logs (ARCRETN).

- ▶ **MAXRTU**

The parameter and setting depend on whether you are using VTS/VE technology or non-virtual tape technology. Even with VTS/VE, which allows for 256 virtual drives, how many virtual drives has your z/OS systems programmer defined? Determine how many drives are available and set aside additional drives for other work. You can override this value by using the SET ARCHIVE command. DB2 allocation can hang if MAXRTU is too low.

- ▶ **TWOARCH**

This parameter “decides” that two software copied versions of the archive will be created (UNIT and UNIT2). Make sure they do not end up on the same physical tape, which defeats the purpose of having a dual copy set.

- ▶ **ARC2FRST**

Depending on how your tape environment was set up, you may need to set ARC2FRST=YES. Determine whether there a duplicate set of dual copied archive logs at the recovery site or only the secondary archive data sets.

## **7.8.4 Large Block Interface**

Starting with DB2 9 NFM, for tape data sets the large block interface is supported (greater than 32760 bytes). This can significantly improve COPY and the restore phase of RECOVER. When used for COPY, the largest block size allowed, which previously was 32,760, is now 256 KB.

The Large Block interface is also used by UNLOAD when the target output is a USS pipe (APAR PK96023).

## **7.8.5 Conclusion**

Tape is an excellent medium to store your DB2 related data. Consider tape for the following DB2 data sets:

- ▶ HSM migrated archive logs
- ▶ Image copies
- ▶ Very large SORTWORK data sets
- ▶ Other assorted data sets

Several challenges exist with using tape in a DB2 environment:

- ▶ VSAM data sets, PDSs and PDSEs must reside on disk.
- ▶ No concurrency or sharing within a data set or volume occurs, therefore parallelism does not exist.
- ▶ Tapes perform best using pure sequential forward access. Most modern tapes do not perform as well when reading backwards (such as for rollback).
- ▶ Data sets residing on tape cannot be striped



# A

## Frequently asked questions

This appendix provides answers to DB2-related frequently asked questions about storage and SMS.

## A.1 What are the components of I/O response time?

The components of I/O response time are as follows:

- ▶ IOSQ time: Queuing at the host/CEC
- ▶ PEND time: Overhead
- ▶ DISC time: Non-productive time
- ▶ CONN time: Data transfer time

### IOSQ time

*IOSQ time* is the time measured when an I/O request is being queued in the LPAR by z/OS.

The following situations can cause high IOSQ time:

- ▶ One of the other response time components is high. When you see a high IOSQ time, look at the other response time components to investigate where the problem actually exists.
- ▶ Sometimes, the IOSQ time is a result of the unavailability of aliases to initiate an I/O request.
- ▶ A slight possibility exists that the IOSQ is caused by a long busy condition during device error recovery.

To reduce the high IOSQ time, perform the following actions:

- ▶ Reduce the other component of the response time. Lowering this other component's response time automatically lowers the IOSQ time.
- ▶ Lower the I/O load through data in memory or use faster storage devices.
- ▶ Provide more aliases. Using HyperPAV is the best option.

### PEND time

*PEND time* represents the time that an I/O request waits in the hardware. This PEND time can be increased as follows:

- ▶ High FICON Director port or DS8000 FICON port utilization. Note the following information:
  - High FICON Director port or DS8000 FICON port utilization can be caused by a high activity rate on those ports.
  - More commonly, high FICON Director port or DS8000 FICON port utilization is a result of daisy chaining multiple FICON channels from separate CECs to the same *port* on the FICON Director or the DS8000 FICON host adapter.

In this case, the FICON channel utilization as seen from the host might be low, but the combination or sum of the utilization of these channels that share the same port (either on the Director or the DS8000) can be significant.
- ▶ High FICON host adapter utilization. Using too many ports within a DS8000 host adapter can overload the host adapter. Use only two out of the four ports in a host adapter.
- ▶ I/O Processor (IOP/SAP) contention at the System z host. More IOP might be needed. IOP is the processor in the CEC that is assigned to handle I/Os.
- ▶ Command reply (CMR) delay, which is a component of PEND time. See Figure 6-1 on page 326. It is the initial selection time for the first I/O command in a chain for a FICON channel. It can be elongated by contention downstream from the channel, such as a busy control unit.
- ▶ Device Busy Delay, also a component of PEND time. See Example 6-1 on page 326. Device Busy Delay is caused by a domain conflict, because of a read or write operation

against a domain that is in use for update. A high Device Busy Delay time can be caused by the domain of the I/O not being limited to the track that the I/O operation is accessing. If you use an independent software vendor (ISV) product, ask the vendor for an updated version, which might help solve this problem.

## **DISC time**

*DISC time* (disconnect or non-productive time) is a measure of the time spent resolving cache misses for READ operations.

If the major cause of delay is the DISC time, search more to find the cause. The most probable cause of high DISC time is having to wait while data is being staged from the DS8000 rank into cache, because of a read-miss operation. This time can be elongated by the following conditions:

- ▶ Low read hit ratio. The lower the read hit ratio, the more read operations have to wait for the data to be staged from the DDMs to the cache. Adding cache to the DS8000 can increase the read hit ratio.
- ▶ High DDM utilization. You can verify high DDM utilization from the ESS Rank Statistics report. Look at the rank read response time. As a general rule, this number must be less than 35 ms. If it is higher than 35 ms, it is an indication that this rank is too busy, because the DDMs are saturated. If the rank is too busy, consider spreading the busy volumes allocated on this rank to other ranks that are not as busy.
- ▶ *Persistent memory full condition* or nonvolatile storage (NVS) full condition. These conditions can also elongate the DISC time.
- ▶ In a Metro Mirror environment, a significant transmission delay between the primary and the secondary site. This condition also causes a higher DISC time.

## **CONN time**

*CONN time* (connect or data transfer time) is the time to transfer the data and is generally considered productive time. It is typically influenced by the size of the blocks being transferred and the utilization of resources.

For each I/O operation, the channel subsystem measures the time that the DS8000, channel, and CEC are connected during the data transmission. When there is a high level of utilization of resources, significant time can be spent in connecting, rather than transferring data. Several reasons for high CONN time are as follows:

- ▶ FICON channel saturation. If the channel or BUS utilization at the host exceeds 50%, it elongates the CONN time. In FICON channels, the data transmitted is divided into frames, and when the channel is busy with multiple I/O requests, the frames from an I/O will be multiplexed with the frames from other I/Os, thus elongating the elapsed time that it takes to transfer all frames that belong to that I/O. The total of this time, including the transmission time of the other multiplexed frames, is counted as CONN time.
- ▶ Contention in the FICON Director, FICON port, and FICON host adapter elongate the PEND time, which also has the same effect on CONN time. Refer to the PEND time discussion in "PEND time" on page 376.
- ▶ Rank saturation caused by high DDM utilization increases DISC time, which also increases CONN time.

## A.2 I hear that DB2 does not work with SMS; is that true?

When SMS was first introduced many years ago there was a recommendation not to use SMS for DB2 data sets. This recommendation was rescinded about two years after SMS was first announced. Therefore, it is not true. You can fully use DB2 with SMS.

## A.3 Is it true that only my DB2 user data, not system data, can be SMS-managed?

The short answer is no. All DB2 data can be SMS-managed. In fact, if you want to use DB2 V8 system-level backup and restore, SMS is required for user and system data sets.

## A.4 If my volumes are SMS-managed, should I worry about space?

Yes. Your likelihood of receiving a call regarding an outage is greatly reduced. However, there are several issues to consider, which are highly dependent on your relationship with your storage administration group:

- ▶ Is the space that you were provided sufficient?
  - Are there any unexpected increases that now make it insufficient?
  - How do you track this now?
  - Who handles these issues, the storage administrator or you?
- ▶ If you are satisfied with the space, is the disk performance sufficient to also satisfy your customer?

## A.5 How do I determine names and available space for SMS-managed volumes?

First, determine your storage group names. Next, determine the volume names by trying the following suggestions:

- ▶ Use ISMF; be sure you are signed in as a storage administrator. Use option 0 for this task.
- ▶ Use option 6, Storage Group, enter the storage group, and then enter LISTVOL for a command.

Example A-1 shows sample output.

*Example A-1 ISMF Storage Group LISTVOL report*

VOLUME	FREE	%	ALLOC	FRAG	LARGEST	FREE
SERIAL	SPACE	FREE	SPACE	INDEX	EXTENT	EXTENTS
-(2)--	---(3)---	(4)-	---(5)---	-(6)-	---(7)---	--(8)--
CBSM01	2598244	94	173256	71	2112449	17
CBSM02	2408773	87	362727	40	2223675	24
CBSM03	2566481	93	205019	39	2327430	16



If you know the VOLSER, you can use ISPF 3.4 with option V. You do not see storage group information, only volume information. See Example A-2.

*Example A-2 ISPF 3.4 volume space listing*

Volume . : CBSM02

Unit . . : 3390

Volume Data	VTOC Data	Free Space	Tracks	Cyls
Tracks . : 50,085	Tracks . : 12	Size . . : 43,530		2,898
%Used . : 13	%Used . . : 12	Largest . : 40,185		2,679
Trks/Cyls: 15	Free DSCBS: 531	Free		
		Extents . : 24		

You can also use DCOLLECT through ISMF or execute IDCAMS. A sample REXX for the output is available in ISMF in **Enhanced ACS Management**  **SMS Report Generation**.

If you have SDSF authority to issue MVS commands, you can display the storage group volumes as shown in Example A-3.

*Example A-3 MVS command to display storage group volumes*

D SMS,SG(CB390),LISTVOL

IGD002I 10:54:27 DISPLAY SMS 404

STORGRP	TYPE	SYSTEM=	1	2	3	4	5	6	7	8
CB390	POOL		+	+	+	+	+	+	+	+

VOLUME	UNIT	SYSTEM=	1	2	3	4	5	6	7	8	STORGRP NAME
CBSM01	9025		+	+	+	+	+	+	+	+	CB390
CBSM02	9034		+	+	+	+	+	+	+	+	CB390
CBSM03	9035		+	+	+	+	+	+	+	+	CB390

\*\*\*\*\* LEGEND \*\*\*\*\*

. THE STORAGE GROUP OR VOLUME IS NOT DEFINED TO THE SYSTEM  
+ THE STORAGE GROUP OR VOLUME IS ENABLED  
- THE STORAGE GROUP OR VOLUME IS DISABLED  
\* THE STORAGE GROUP OR VOLUME IS QUIESCED  
D THE STORAGE GROUP OR VOLUME IS DISABLED FOR NEW ALLOCATIONS ONLY  
Q THE STORAGE GROUP OR VOLUME IS QUIESCED FOR NEW ALLOCATIONS ONLY  
> THE VOLSER IN UCB IS DIFFERENT FROM THE VOLSER IN CONFIGURATION  
SYSTEM 1 = SYSA      SYSTEM 2 = SYSB      SYSTEM 3 = SYSC  
SYSTEM 4 = SYSD      SYSTEM 5 = SYSE      SYSTEM 6 = SYSF  
SYSTEM 7 = SYSG      SYSTEM 8 = SYSPLEX1

## A.6 How many storage groups should I have in my production environment?

Production must have its own storage group for DB2 table spaces and indexes that are separate from all other environments.

Consider the following approaches:

- ▶ Start with at least three separate storage groups. If you will be using the DB2 V8 BACKUP and RESTORE SYSTEM (which requires z/OS 1.5) you are required to have separate Storage groups for the BSDS and active log data sets from all other data. The three categories are as follows:
  - DB2 catalog and directory objects
  - BSDS and active log data sets
  - DB2 user data
- ▶ Discuss ICF catalog placement strategies with your storage administrator regarding various recovery scenarios.
- ▶ Place the sort (DSNDB07) data sets on a separate volume from those volumes if not using PAV and MA.

Newer disk devices that use PAV, MA, and so forth typically do not require data and indexes on separate volumes. However, older versions might benefit by the separation (however, continue to watch for hot spots). For separate storage groups, use the following methods:

- Use a unique data set naming convention separating data and indexes that will be resolved by the storage group ACS routine for correct placement.
  - Use ZPARM values for SMSDCFL and SMSDCIX, in macro DSN6SPRM. These hidden DSNZPARM values provide SMS with one Data Class for data (SMSDCFL) and another for indexes (SMSDCIX) that will be resolved by the storage group ACS routine for correct placement.
- ▶ Provide a storage group for your archive log data sets.

Recovery using archive log from disk is much faster than tape for parallel recoveries where several logs reside on the same tape. The exception to this is when archive logs are stored on DFSMSHsm tapes (aside from recall time). In this case, make sure the storage group can handle additional logs.
  - ▶ Provide a storage group for image copy data sets. Recoveries from image copy data sets residing on disk is much faster for parallel operations because there is no need to serialize image copy data sets if stacked on the same tape.
  - ▶ Determine realistically how many archive log and image copy data sets are required for recovery situations, and size the volumes in your storage groups accordingly.

## A.7 How many storage groups should I have in non-production?

For non-production, the number depends on the requirements for the following items:

- ▶ Performance
- ▶ Backup and recovery
- ▶ Types of environments: sandbox, development, test, ERP and non-ERP, and so on
- ▶ Amount of data in each environment

Separation of environments depends on business requirements. You might want to stay with the production strategy, or you might want to combine some or all of the storage groups for the these environments.

## A.8 What are considerations for consolidating to large volumes: Mod 27 or 54

As an example, if your storage administrator is putting in mod 54s for application data. Instead of your current 15 mod 3 volumes, the storage administrator is trading up to one volume with the capacity of a little bit more than 19 mod 3s. If that sounds like a lot, consider the following suggestions:

- ▶ Although VSAM data sets can have up to 255 extents (increased in z/OS 1.7), there is a limitation of 123 extents per volume. This limitation means that extending past 123 extents is not possible with only one volume, so you will never grow beyond this point. This is true for other object types also, such as extended format sequential data sets, which, similar to VSAM, allow 123 extents per volume.
- ▶ If you are currently backing up (full volume dump) your volumes, a lack of parallelism exists. Only one very large dump occurs instead of up to 20 dumps in parallel. Determine how long this function takes, which is true for other operations, such as DEFRAG.
- ▶ Consider the amount of time it takes to FlashCopy a volume. Again, similar to the dump issue, your copies will not be in parallel.

## A.9 How did these space allocations happen?

Consider the following scenarios:

- ▶ In the first scenario, you created an image copy data set as `SPACE=(CYL,(1000,100))` and expected 1400 cylinders and five extents when it went to multi-volume, but received 2300 cylinders with five extents. What happened?

If your data set uses the guaranteed space attribute, SMS is working as designed. In this example the primary allocation is propagated to the next volume when it went to multi-volume. Extents are as follows: `VOL1=1000,100 VOL2=1000,100,100`.

- ▶ In the second scenario, you image-copied an 800-track table space by mistake. The output was `TRK(1,1)`, but the allocation worked. What happened?

If your storage administrator assign EF for your image copy data sets and if there are at least seven volumes with enough free space, the data set can spread up to 123 extents (as opposed to 16 extents non-EF) on each volume, the end result will be a sequential data set with 800 extents. This is not a great way of doing business, but there is no outage. With z/OS 1.7 you can have roughly 7257 extents max for 60 volumes. Extent consolidation with SMS is also a possibility.

## A.10 My storage administrator sees many disk-write bursts. How can I help?

Your storage administrator sees many disk-write bursts in an RMF Cache Volume Detail report as “DFW Bypass.” For newer disks, this bypass is actually a DASD Fast Write retry and no longer a bypass.

This message means that non-volatile storage (NVS) is being flooded with too many writes operations. The controller retries the write operation in anticipation that some data in NVS has been off-loaded.

For RAMAC devices, the solution was to lower VDWQT to 0 or 1:

- ▶ This solution caused high DBM1 SRM CPU time.
- ▶ This solution might no longer be needed for ESS and DS8000 devices. Test and verify the settings.

For very large buffer pools with many data sets, consider lowering VDWQT to 0 or 1:

- ▶ This solution also might work well for ESS and DS8000. The trade-off is still higher DBM1 SRM CPU time.
- ▶ Test and retest. Validate such things as Class 3 times.

## A.11 Do I need PDSEs?

I have heard about PDSEs. What are they, and do I need them? Do they have to be SMS-managed?

PDSEs do not have to be SMS-managed (partial APAR list - OW39951).

Before DB2 9, there were no requirements for PDSEs. The following items are delivered as PDSEs starting with DB2 9:

- ▶ ADSNLOAD
- ▶ ADSNLOD2
- ▶ SDSNLOAD
- ▶ SDSNLOD2

Use PDSEs for load libraries that contain stored procedures. This reduces the risk of out-of-space conditions because of adding or updating members. This is true for DB2 V8 too.

PDSEs are like PDSs, but offer more:

- ▶ Up to 123 extents are allowed instead of 16 for PDSs. Cannot extend beyond one volume.
- ▶ Number of directory blocks is unlimited.
- ▶ Does not require periodic compression to consolidate fragmented space for reuse.
- ▶ No need to re-create PDSEs when the number of members expands beyond the available directory blocks of PDS.

## A.12 Why do I have space anomalies?

This question has two scenarios: Reasons for the space anomalies and having less space than requested.

### **Reason for the space anomalies**

In this scenario, you have 90 cylinders primary, 12 cylinders secondary, 63 extents, but 25530 tracks allocated. See Example A-4.

*Example A-4 Report of space anomalies: Question 1*

---

```
ISPF 3.4 listing                      Tracks  XT  Device
-----
RI1.DSNDBD.A020X7KR.CKMLPR.I0001.A001  25530  63  3390

LISTCAT output:
ALLOCATION
      SPACE-TYPE-----CYLINDER      HI-A-RBA-----1254850560
      SPACE-PRI-----90      HI-U-RBA-----1113292800
      SPACE-SEC-----12
```

primary+(secondary\*(extents-1))=space  
90+(12\*(63-1))=834 cylinders, 12510 tracks, not 25530!

---

The sequence of events that led to the anomaly are as follows:

1. CREATE TABLESPACE PRIQTY 64800 SECQTY 8640 was issued.
2. After time and space, ALTER TABLESPACE SECQTY 20000 was issued.
3. More rows and trip extents were added.
4. No REORG operation was performed afterward.
5. The results indicated that the DB2 information was correct, but MVS information was not.
6. CYL(90,12) with 63 extents should be 2510 tracks. The exception to this is when you have additional extents because of volume fragmentation.

Discuss with your storage administrator that what is seen might not be accurate.

This case is not a disk fragmentation issue. After the ALTER process, DB2 knows the allocation converted to CYL(90,28). However, MVS still thinks it is CYL(90,12) until redefined by a process such as REORG without a REUSE. PRIQTY and SECQTY is actually what DB2 uses, not CYL(90,12).

## Having less space than requested

How did you actually get *less* space than you requested?

When your storage administrator has set up a Data Class with the space constraint relief attribute on, and with the request for a percentage for space reduction, your allocated data set can actually be less than requested. This can happen if your volume does not have enough space.

For example, assume that a 4 KB object was created with PRIQTY 72000 (100 cylinders), the Data Class space constraint was set up to allow 10% reduction, you had one volume with 92 cylinders remaining.

Results are as follows:

- ▶ The DB2 catalog still shows the equivalent of PRIQTY 72000.
- ▶ The actual MVS allocation will be 90 cylinders or the equivalent of PRIQTY 64800.

## A.13 What about catalogs?

When storage administrators talk about *catalogs*, are they talking about the *DB2 catalog*?

Generally, the answer here is no. storage administrators view the term catalog as the ICF catalog, which they typically maintain. Make sure that when you or your storage administrator use the term catalog it is specifically stated which one. This distinction avoids needless confusion, errors, and arguments.

## A.14 Do I need to let my storage administrator know anything about DB2 V8?

If you are migrating DB2 to V8, what should you tell your storage administrator?

The information you provide depends on how your storage administrator set up the ACS routines for the DB2 data set names. If the storage administrator is looking at the low-level qualifier of your DB2 data set name and you plan to use partitions above the V7 limit of 254, the answer is yes.

The LLQ in DB2 V8 have the following pattern:

- ▶ A001-A999 for partitions 1 - 999
- ▶ B000-B999 for partitions 1000 - 1999
- ▶ C000-C999 for partitions 2000 - 2999
- ▶ D000-D999 for partitions 3000 - 3999
- ▶ E000-E096 for partitions 4000 - 4096

Your storage administrator might need to change certain reporting programs also.

## A.15 What is extent reduction?

My storage administrator told me that my DB2 data sets were re-created with high extents. They are now as low as one extent per data set. Are there any issues?

Storage administrators have several ways of causing extent reduction that potentially bring back a data set in one extent:

- ▶ DFSMSHsm MIGRATE and RECALL functions
- ▶ DFSMSdss COPY or DUMP and RESTORE functions
- ▶ DEFRAg with the CONSOLIDATE keyword

Using functions such as DFSMSdss COPY might be faster than running REORGs. Perhaps you have SQL that uses the DB2 catalog to report on extents, which can potentially be a problem. You might be redoing REORGs unnecessarily because the move was done outside of DB2 and DB2 does not recognize it.

**Note:** The EXTENTS column in the RTS is updated only when an update or applicable utility is run for the object. A simple start after extent reduction or a read based on SELECT will not update the EXTENTS column (same issue as the catalog).

Using high extents as a tool to review issues with clustering indexes can potentially be a problem. Review CLUSTERRATIOF more closely.

## A.16 How much data can I really lose?

Is it possible to lose only one or a few volumes with newer disk? Yes, although losing one or several volumes instead of an entire disk controller's worth is extremely rare.

Because of this possibility (there are other possibilities), if you run a daily disk report by volume for your DB2 objects, consider the following questions if something does go wrong:

- ▶ What was on the lost volume?
  - Did you lose all indexes? Perhaps you can rebuild them.
  - If part of one application or part of a partition data set, it might not be too bad.
  - Was it your new mod 54 with *all* of your data? Determine what your alternatives are. Based on the architecture you have built in for this type of event, you should have alternatives. Otherwise, the answer gets into a more detailed analyzation beyond the scope of this book.
- ▶ Should I care whether my data sets are multi-volume? Yes, based on the information presented here. Lots of multi-volume data sets can cause you lots of additional restore operations if a piece of your data resides on the failed volume.

## A.17 What about pseudo deletes?

My index keeps growing and tripping extents, even after deletes. What is wrong with DB2? How can I control the extent growth?

This issue is actually a DB2 issue concerning pseudo deleted entries in your index, not a Storage Management issue:

- ▶ For an index, deleted keys are marked as pseudo deleted.
- ▶ Actual cleaning up does not occur except during certain processes (for example, before a page split).
- ▶ High CPU cost of an index scan can be involved. Each time an SQL statement makes a scan of an index, it has to scan all entries in the index, including pseudo deleted entries that have not yet been removed.
- ▶ You can calculate the percentage of RIDs that are pseudo deleted based on the values of PSEUDO\_DEL\_ENTRIES and CARDF in SYSINDEXPART:  
$$(PSEUDO\_DEL\_ENTRIES / CARDF) * 100$$

Use REORG INDEX, if the percentage of pseudo deleted entries is greater than 10% for non Data Sharing and 5% for Data Sharing.

## A.18 What are FlashCopy and SnapShot?

The difference between SnapShot and FlashCopy versions 1 and 2 (at a high level) are as follows:

- ▶ SnapShot (RVA only)
  - SnapShot can quickly move data from the source device to the target device.
  - Data is “snapped” (quickly copied) directly from the source location to the target location.
- ▶ FlashCopy (ESS and DS8000) versions 1 and 2
  - FlashCopy V1 requires the entire source volume and target volume to be involved in a FlashCopy relationship. FlashCopy V1 relationships do not allow any other FlashCopy relationships to exist on either the source or target volume.
  - FlashCopy V2 enhances the FlashCopy function by providing an alternative method to copying an entire source volume to a target volume:
    - Multiple FlashCopy relationships are allowed on a volume.
    - Track relocation is possible because when copying tracks, the target tracks do not have to be in the same location on the target volume as on the source volume.
    - The restriction that a FlashCopy target and source volume must be in the same logical subsystem (LSS) in an ESS is removed. However, FlashCopy must still be processed in the same ESS.

V2 is 10 times faster than FlashCopy V1.

## A.19 How many buffer pools do I need?

Besides those needed for the DB2 catalog and directory (BP0, BP8K, BP16K0 and BP32K0), the standard is to allocate your work files to dedicated buffer pools. Most customers name this buffer pool BP7. At a minimum, split your user application indexes and table spaces into separate buffer pools of their own.

You might improve the performance of I/O operations by increasing the size of your buffer pools. Make buffer pools as large as you can afford for the following reasons:

- ▶ Using larger buffer pools might mean fewer I/O operations and therefore faster access to your data.
- ▶ Using larger buffer pools can reduce I/O contention for the most frequently used tables and indexes.
- ▶ Using larger buffer pools can speed sorting by reducing I/O contention for work files.



## A.20 What guidelines exist for sizing structures in the CF regarding the GBP?

There are three aspects of group buffer pool (cache structure) size to consider:

- ▶ Total structure size: The total structure size of a group buffer pool is specified in the coupling facility policy definition for the cache structure.
- ▶ Number of directory entries: A directory entry is used by the coupling facility to determine where to send cross-invalidation signals when a page of data is changed or when that directory entry must be reused. A directory entry contains control information for one database page, no matter in how many places that page is cached. For example, if page P1 is cached in the group buffer pool and in the buffer pools of three members, that page still has only one directory entry.
- ▶ The number of data entries: Data entries are the actual places where the data page resides. These are 4 KB, 8 KB, 16 B, or 32 KB in size (the same size as the data page). For GBPCACHE NO group buffer pools, no data entries exists.

## A.21 What is a good way to establish a buffer pool strategy?

Consider using the AUTOSIZE option, which was introduced with DB2 9 for z/OS.

The AUTOSIZE allows DB2 to increase or decrease the size of a given buffer pool by up to 25% of the originally allocated size.

It can be activated with a new AUTOSIZE(YES) option on the ALTER BUFFERPOOL command.

After activation, it can be deactivated as follows:

```
ALTER BUFFERPOOL (bpname) AUTOSIZE(NO)
```

The AUTOSIZE attribute is added to the DISPLAY BUFFERPOOL output.



# Abbreviations and acronyms

<b>48 KB</b>	49,152 bytes	<b>DFDSS</b>	Data Facility Data Set Services
<b>AAL</b>	arrays across loops	<b>DFHSM</b>	Data Facility Hierarchical Storage Manager
<b>ABARS</b>	aggregate backup and recovery support	<b>DFP</b>	Data Facility Product
<b>ACDS</b>	active control data set	<b>DFS</b>	distributed file system
<b>ACS</b>	automatic class selection	<b>DFW</b>	DASD Fast Write
<b>AMP</b>	Adaptive Multi-stream Prefetching	<b>DGTT</b>	declared global temporary table
<b>API</b>	application programming interface	<b>DHCP</b>	Dynamic Host Configuration Protocol
<b>ASCII</b>	American Standard Code for Information Interchange	<b>DMA</b>	direct memory access
<b>ASIC</b>	application-specific integrated circuit	<b>DMZ</b>	De-Militarized Zone
<b>ATL</b>	automated tape library	<b>DNS</b>	Domain Name System
<b>BBU</b>	battery backup unit	<b>DPR</b>	Dynamic Path Reconnect
<b>BCDS</b>	backup control data set	<b>DPS</b>	Dynamic Path Selection
<b>BLOB</b>	binary large object	<b>DSCB</b>	data set control block
<b>BPV</b>	break point value	<b>DSS</b>	Data Set Services
<b>BSDS</b>	bootstrap data set	<b>DVC</b>	Dynamic Volume Count
<b>CA</b>	control area	<b>DVE</b>	Dynamic Volume Expansion
<b>CEC</b>	central electronics complex	<b>DWQT</b>	deferred write threshold
<b>CG</b>	Consistency Group	<b>EA</b>	extended addressability
<b>CGTT</b>	created global temporary tables	<b>EAV</b>	extended address volume
<b>CHPID</b>	Channel Path ID	<b>EB</b>	exabyte
<b>CI</b>	control interval	<b>ECC</b>	error checking and correction
<b>CIDF</b>	control interval definition field	<b>EDF</b>	Extended Distance FICON
<b>CKD</b>	count key data	<b>EEH</b>	Enhanced Error Handling
<b>CLI</b>	command-line interface	<b>EF</b>	Extended Format
<b>CLOB</b>	character large object	<b>ESCON</b>	Enterprise Systems Connection
<b>CM</b>	conversion mode	<b>ESS</b>	Enterprise Storage Server
<b>CMR</b>	command reply	<b>EU</b>	End User
<b>COMMDS</b>	communications data set	<b>FATA</b>	Fibre ATA
<b>CPU</b>	central processing unit	<b>FB</b>	fixed block
<b>CSDO</b>	Certified Secure Data Overwrite	<b>FCP</b>	Fibre Channel Protocol
<b>CoD</b>	Capacity on Demand	<b>FCSE</b>	FlashCopy Space Efficient
<b>CXRC</b>	coupled remote extended copy	<b>FDE</b>	Full Disk Encryption
<b>DA</b>	device adapter	<b>FFDC</b>	First Failure Data Capture
<b>DASD</b>	direct access storage device	<b>FICON</b>	Fiber Connection
<b>DB2</b>	DB2 for z/OS	<b>FIT</b>	fast implementation techniques
<b>DBA</b>	data base administrator	<b>FLA</b>	Fast Log Apply
<b>DCB</b>	data control block	<b>FRR</b>	Failure Recovery Routines
<b>DDM</b>	disk drive module	<b>FTP</b>	File Transfer Protocol
		<b>GB</b>	gigabyte

<b>GC</b>	Global Copy	<b>KB</b>	kilobyte
<b>GDG</b>	generation data set group	<b>Kb</b>	kilobit
<b>GM</b>	Global Mirror	<b>Kbps</b>	kilobits per second
<b>GRS</b>	global resource serialization	<b>LBA</b>	logical block address
<b>GSA</b>	Global Storage Architecture	<b>LC19</b>	log latch
<b>GUI</b>	graphical user interface	<b>LCU</b>	logical control unit
<b>HA</b>	Host Adapter	<b>LDAP</b>	Lightweight Directory Access Protocol
<b>HACMP™</b>	High-Availability Cluster Multi-Processing	<b>LDS</b>	linear data set
<b>HBA</b>	host bus adapter	<b>LED</b>	light emitting diode
<b>HCD</b>	hardware configuration definition	<b>LFU</b>	least frequently used
<b>HDA</b>	head disk assembly	<b>LIP</b>	Loop initialization primitive
<b>HDD</b>	hard disk drive	<b>LLQ</b>	low level qualifier
<b>HLL</b>	high level language	<b>LMC</b>	Licensed Machine Code
<b>HLQ</b>	high-level qualifier	<b>LOB</b>	large object
<b>HMC</b>	Hardware Management Console	<b>LOB</b>	large data object
<b>HSM</b>	Hardware Security Module	<b>LPAR</b>	logical partition
<b>HTTP</b>	Hypertext Transfer Protocol	<b>LRU</b>	least recently used
<b>HTTPS</b>	Hypertext Transfer Protocol over SSL	<b>LSS</b>	logical subsystem
<b>IBM</b>	International Business Machines Corporation	<b>LUN</b>	logical unit number
<b>IFC</b>	instrumentation facility component	<b>LVM</b>	Logical Volume Manager
<b>IFCID</b>	instrumentation facility component identifier	<b>MA</b>	Multiple Allegiance
<b>IFI</b>	instrumentation facility interface	<b>MB</b>	megabyte
<b>IKE</b>	Internet Key Exchange	<b>MCU</b>	multicylinder unit
<b>IKS</b>	Isolated Key Server	<b>MFU</b>	most frequently used
<b>IOP</b>	I/O processor	<b>MGM</b>	Metro Global Mirror
<b>IOCDS</b>	I/O configuration data set	<b>MIB</b>	Management Information Base
<b>IODF</b>	I/O definition file	<b>ML1</b>	migration level 1
<b>IOPS</b>	I/O operations per second	<b>ML2</b>	migration level 2
<b>IOS</b>	I/O supervisor	<b>MM</b>	Metro Mirror
<b>IOSQ</b>	IOS queue	<b>MPIO</b>	multipath I/O
<b>IOSQD</b>	IOSQ delay	<b>MRU</b>	most recently used
<b>IPL</b>	initial program load	<b>MSS</b>	Multiple Subchannel Sets
<b>IPSec</b>	Internet Protocol Security	<b>Mb</b>	megabit
<b>IPv6</b>	Internet Protocol version 6	<b>Mbps</b>	megabits per second
<b>ISMF</b>	Interactive Storage Management Facility	<b>NAT</b>	Network Address Translation
<b>ITSO</b>	International Technical Support Organization	<b>NFM</b>	new-function mode
<b>IWC</b>	Intelligent Write Caching	<b>NFS</b>	Network File System
<b>IWTH</b>	immediate write threshold	<b>NIP</b>	Nucleus Initialization Program
<b>JCL</b>	job control language	<b>NVRAM</b>	Non-Volatile Random Access Memory
<b>JFS</b>	Journaling File System	<b>NVS</b>	nonvolatile storage
		<b>OEL</b>	Operating Environment License
		<b>OLTP</b>	online transaction processing

<b>PATA</b>	Parallel Attached Technology Adapter	<b>SMP</b>	Symmetric Multiprocessor
<b>PAV</b>	parallel access volume	<b>SMS</b>	System Managed Storage
<b>PB</b>	petabyte	<b>SPE</b>	Small Programming Enhancement
<b>PCM</b>	Path Control Module	<b>SPTH</b>	sequential prefetch threshold
<b>PCU</b>	physical control unit	<b>SRM</b>	Storage Resource Management
<b>PDS</b>	partitioned data set	<b>SSD</b>	solid-state drive
<b>PDSE</b>	partitioned data set extended	<b>SSID</b>	Subsystem Identifier
<b>PFA</b>	Predictive Failure Analysis	<b>SSL</b>	Secure Sockets Layer
<b>PIT</b>	point-in-time	<b>SSPC</b>	System Storage Productivity Center
<b>PM</b>	Preserve Mirror	<b>SSRE</b>	System Services Runtime environment
<b>PMB</b>	Physical Memory Block	<b>STCK</b>	store clock
<b>PPRC</b>	Peer-to-Peer Remote Copy	<b>SVC</b>	SAN Volume Controller
<b>PPRC-XD</b>	Peer-toPeer Remote Copy Extended Distance	<b>TB</b>	terabyte
<b>PTC</b>	point-in-time copy	<b>TCE</b>	Translation Control Entry
<b>RAID</b>	Redundant Array of Independent Disks	<b>TCO</b>	total cost of ownership
<b>RAM</b>	random access memory	<b>TCP/IP</b>	Transmission Control Protocol / Internet Protocol
<b>RAS</b>	reliability, availability, serviceability	<b>TVC</b>	tape volume cache
<b>RIO</b>	remote input/output	<b>UCB</b>	unit control block
<b>RMF</b>	Resource Measurement Facility	<b>UDID</b>	Unit Device Identifier
<b>RMM</b>	removable media manager	<b>UTS</b>	universal table spaces
<b>RPC</b>	rack power control	<b>VC</b>	Volume Count
<b>RPM</b>	revolutions per minute	<b>VDWQT</b>	vertical deferred write threshold
<b>RPO</b>	recovery point objective	<b>VMA</b>	Volume Mount Analyzer
<b>RPQ</b>	request for price quotation	<b>VPN</b>	Virtual Private Network
<b>SA</b>	storage administrator	<b>VSAM</b>	Virtual Storage Access Method
<b>SAN</b>	storage area network	<b>VTOC</b>	volume table of contents
<b>SAP</b>	system assist processor	<b>VTG</b>	Virtual Tape Server
<b>SARC</b>	Sequential Adaptive Replacement Cache	<b>WLM</b>	Workload Manager
<b>SATA</b>	Serial Attached Technology Adapter	<b>XRC</b>	extended remote copy
<b>SCDS</b>	source control data set	<b>YB</b>	yottabyte
<b>SCR</b>	Space Constraint Relief	<b>ZB</b>	zettabyte
<b>SCSI</b>	Small Computer System Interface	<b>zHPF</b>	High Performance FICON for z
<b>SDD</b>	subsystem device driver	<b>zAAP</b>	System z Application Assist Processor
<b>SDM</b>	system data mover	<b>zIIP</b>	System z Integrated Information Processor
<b>SDR</b>	sustained data rate		
<b>SE</b>	storage enclosure		
<b>SFI</b>	Storage Facility Image		
<b>SFTP</b>	SSH File Transfer Protocol		
<b>SMF</b>	System Management Facilities		
<b>SMIS</b>	Storage Management Initiative Specification		



# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

## IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 395. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *IBM System Storage Solutions Handbook*, SG24-5250-07
- ▶ *ABCs of z/OS System Programming Volume 3*, SG24-6983
- ▶ *DB2 for OS/390 and Data Compression*, SG24-5261
- ▶ *DB2 9 for z/OS: Backup and Recovery I/O Related Performance Considerations*, REDP-4452
- ▶ *DB2 9 for z/OS: Buffer Pool Monitoring and Tuning*, REDP-4604
- ▶ *DB2 UDB for z/OS Version 8: Everything You Ever Wanted to Know, ... and More*, SG24-6079
- ▶ *DB2 UDB for z/OS Version 8 Performance Topics*, SG24-6465
- ▶ *DFSMSHsm Fast Replication Technical Guide*, SG24-7069
- ▶ *DFSMS V1.10 and EAV Technical Guide*, SG24-7617
- ▶ *Disaster Recovery with DB2 UDB for z/OS*, SG24-6370
- ▶ *DS8000 Performance Monitoring and Tuning*, SG24-7146
- ▶ *Effective zSeries Performance Monitoring Using Resource Measurement Facility*, SG24-6645
- ▶ *How does the MIDAW Facility Improve the Performance of FICON Channels Using DB2 and other workloads?*, REDP-4201
- ▶ *IBM System Storage DS8000: Architecture and Implementation*, SG24-6786
- ▶ *IBM System Storage DS8000: Remote Pair FlashCopy (Preserve Mirror)*, REDP-4504
- ▶ *IBM System Storage DS8000 Series: IBM FlashCopy SE*, REDP-4368
- ▶ *IBM Virtualization Engine TS7700 with R1.6*, SG24-7712-01
- ▶ *Index Compression with DB2 9 for z/OS*, REDP-4345
- ▶ *Multiple Subchannel Sets: An Implementation View*, REDP-4387
- ▶ *Optimizing Restore and Recovery Solutions with DB2 Recovery Expert for z/OS V2.1*, SG24-7606
- ▶ *Ready to Access DB2 for z/OS Data on Solid-State Drives*, REDP-4537

## Other publications

These publications are also relevant as further information sources:

- ▶ *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840-05
- ▶ *DB2 Version 9.1 for z/OS Command Reference*, SC18-9844-04
- ▶ *DB2 Version 9.1 for z/OS Installation Guide*, GC18-9846-08
- ▶ *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851-06
- ▶ *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854
- ▶ *DB2 Version 9.1 for z/OS Utility Guide and Reference*, SC18-9855-04
- ▶ *DFSMS/MVS DFSMSdfp Storage Administration Reference*, SC26-7402
- ▶ *DFSMS/MVS DFSMSdfp Storage Administration Reference*, SC26-7402
- ▶ *DFSMS/MVS V1R3 NaviQuest User's Guide*, SC26-7194
- ▶ *DS8000 Copy Services for IBM System z*, SG24-6787
- ▶ *Tivoli OMEGAMON XE for DB2 Performance Expert and Performance Monitor on z/OS Configuration and Customization*, V4R2, GC19-2511
- ▶ *z/Architecture Principles of Operations*, SA22-7832
- ▶ *z/OS DFSMS Advanced Copy Services*, SC35-0428
- ▶ *z/OS Resource Measurement Facility Report Analysis*, SC33-7991
- ▶ *z/OS V1R10.0-V1R11.0 DFSMS Introduction*, SC26-7397
- ▶ *z/OS V1R11.0 DFSMS Implementing System-Managed Storage*, SC26-7407
- ▶ *z/OS V1R11.0 DFSMS Managing Catalogs*, SC26-7409
- ▶ *z/OS V1R11.0 DFSMS Using Data Sets*, SC26-7410
- ▶ *z/OS V1R11.0 DFSMS Using the Interactive Storage Management Facility*, SC26-7411
- ▶ *z/OS V1R11.0 DFSMS Using the New Functions*, SC26-7473-06
- ▶ *z/OS V1R12.0 DFSMS Using the New Functions*, SC26-7473-07

## Online resources

These websites are also relevant as further information sources:

- ▶ DS8000 Terms and Concepts  
<http://www.redbooks.ibm.com/abstracts/tips0535.html?Open>
- ▶ System z hardware  
<http://www.ibm.com/systems/z/hardware/>
- ▶ Tivoli OMEGAMON XE for Storage on z/OS  
<http://www.ibm.com/software/tivoli/products/omegamon-xe-storage/>
- ▶ IntelliMagic Vision  
<http://www.intellimagic.net/intellimagic/products/rmf-magic>



## How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks publications, at this web site:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Help from IBM

IBM Support and downloads

[ibm.com/support](http://ibm.com/support)

IBM Global Services

[ibm.com/services](http://ibm.com/services)



# Index

## Numerics

3380 15, 34, 152, 214  
3390 15, 134–135, 214, 227–228, 245, 379  
3390 Model A 18  
3953 105

## A

AAL 39–40  
ABARS 199–200, 203, 212, 230, 269  
ACDS 204  
ACS 5, 141, 184, 187, 193, 196, 203, 205, 219, 221, 249, 299, 379  
ACS routines 132, 184, 187, 205–207, 221–222, 261, 280, 300–301, 380, 384  
    actual size 231  
    assigning storage classes 210  
    data sets 308  
    device allocations 261  
    for allocation 187  
    permit user definition 206  
    source code 299  
ACS test  
    case 300  
    case, Alter 300  
    library 234, 300  
    member 300  
    selection 300  
active control data set (ACDS) 204  
active log 147–148, 204, 212, 216, 230–231, 233, 235, 260–261, 265, 324, 340, 351, 380  
    duplex pair 289  
    sizing 148  
active log data sets 238, 261–262  
    default names 174  
Adaptive Multi-stream Prefetching  
    *See* AMP  
ADDVOL 198–200  
affinity 48, 52  
AIX 43, 46  
alias address 26–27, 326  
alias devices 29  
allocation 3, 21, 51, 55, 113, 121, 128, 200, 202, 221–222, 244, 381  
allocation unit 51, 182  
allocation/migration threshold 271  
AMP 6, 66, 149  
Arbitrated Loop Physical Addressing (AL-PA) 48  
ARC0640I ARCFRTM 167  
architecture 2, 11, 15, 146, 385  
ARCHIVE 150, 265  
archive log 9, 141, 144, 147–150, 197, 200, 213, 226, 229–231, 239, 261, 323, 340, 380  
    recovery requirement 312  
    Storage Group 216

    swept hourly 216  
archive log data set 150, 213, 229, 238, 265–266, 362  
    default names 175  
archive to disk or tape 362  
array 35  
array site 39, 48–49  
    eighth disk 39  
arrays 23, 36, 40, 49, 149–150  
arrays across loops *See* AAL  
asynchronous write 351–352  
AUTO BACKUP 213, 215  
automated space management 177  
Automated Tape Library (ATL) 8, 34  
automatic class selection  
    *See* ACS  
availability management 3, 198, 200, 203, 213  
Avg Value 240  
AVGLRECL 223, 250

## B

backup xix, 2, 6–7, 34, 136, 148, 194, 198, 200, 205, 230, 269, 339, 378, 380  
    automatic 213  
    storage group type 274  
    system copies 160  
    V8 380  
backup control data set (BCDS) 161  
BACKUP System 161  
base address 26, 326  
    I/O activity 29  
base addressing space 20  
base configuration 204, 207, 290  
benefits of SMS 3, 202, 215  
BLANK DSCBS 303  
block size 21, 113, 122, 203, 227–228, 242, 244  
bootstrap data set  
    *See* BSDS  
break point value (BPV) 217, 273  
BSDS 147–148, 293–294  
    default names 174  
    definition of 147  
buffer pool 2, 124, 139, 146, 153, 202, 318, 321, 339–340, 348, 352, 382  
    several consecutive pages 344  
    total size 348  
    unavailable pages 348  
    updated pages 349  
BUFFERPOOL BP0 128  
BUFSPACE 223, 250  
BUS utilization 328, 377  
bypass cache 5  
Byte EA 120

## C

### CACHE

- display 290
- utility option 164
- cache 2, 5, 36, 43–44, 113, 149, 188, 202, 210, 340, 344
  - available 238
  - hit 340
  - multiple misses 36
  - performance 210
  - volume report 381
- candidate volume 208–209, 252, 312
- Capacity on Demand 34
- catalog xix, 20, 126, 136, 195, 217, 219, 230, 380
- CDS Name 220, 240
- channel program 26, 32, 330
- CI 122
  - free space 242
  - size 122, 212, 223, 242, 247
  - size, KB page 122
  - VSAM 122
- CKD volume 53–55
  - distribute I/O load to 57
  - maximum size 53
  - minimum physical allocation 51
- cluster 122, 128, 220–221, 243, 308, 344
- clustering indexes 384
- communications data set (COMMDS) 204
- components 8, 35, 44, 145, 193–194
- compression xix, 124, 145–146, 209, 267, 339, 382
- concurrent copy 81, 157, 213, 258, 260
  - capability 310
  - feature 157
- conditional restart control record (CRCR) 168
- configuration
  - volume 16, 215, 230, 289
- CONN time 326, 376–377
  - elongate 377
  - same effect 377
- Consistency Group FlashCopy 89
- constraint 22, 152, 383
- Control Data Set (CDS) 196, 289
- control interval
  - See CI
- CONVERT 197
- converting to SMS 197
- COPY 167, 196, 381
  - function 384
  - table space 227, 272
  - tablespace 133
- copy pool 160, 196
  - backup SG name 271, 275
  - backup SG type 276
  - backup storage group 162, 273
  - FAST REPLICATION BACKUP 167
  - fast replication backup 167
  - fast replication output 167
  - FAST REPLICATION RECOVERY 170
  - type 166
  - type use DB 166
- copy service 14

- summary list 14
- COPY utility 116
- count field 112
- count-key-data (CKD) 26
- CPU cost 146, 385
- CPU time 344, 382
- CREATE DATABASE 124, 137
- CREATE INDEX 124, 128
- CREATE STOGROUP 126, 140, 220–221, 303
- CREATE TABLESPACE 122–123, 127, 221, 223, 303
- created global temporary tables (CGTT) 139
- cylinder (cyl) 121, 129
- cylinder-managed space 20–21, 215

## D

- DA 39–40, 45, 47, 155
  - device interface 47
- DA pair 40
  - first disk loop 40
  - second disk loop 40
- DASD fast write (DFW) 330
- DASD volume 13–14, 152
- data base
  - administrator 310
- Data Class 126, 132, 194–195, 205, 239, 264, 380
  - default information 250
- data compression 146
  - Additional information 146
  - many differences 147
- Data Facility Hierarchical Storage Manager
  - See DFHSM
- Data Facility Product (DFP) 195
- data locality 5
- data page 340
- data placement 194
- data set xix, 2–3, 5, 17, 115–116, 184, 195–196, 219, 229, 240, 249, 324, 349, 380–381, 383–385
  - different types 3
  - exclusive use 314
  - high availability 310
  - ISPF 3.4 245
  - many different options 208
  - maximum number 138
  - maximum size 138
  - new types 133
  - new z/OS support 152
  - particular type 125
  - physical deletion 160
  - proper management 116
  - sample SMS constructs 237
  - service level 210
  - service requirements 314
  - updated pages 352
- data sharing 125, 340, 385
- data spaces 341
- database xix, 2–3, 116–117, 119, 125, 193, 213, 215, 232
- DATACLAS 245
- DATE column 168
- DB2 xix, 2, 11, 115, 193–194, 237, 298, 317, 339

- accounting trace 351
- administrator 3, 148, 203, 310
- DB2 9 6, 116, 220, 257, 339–340, 382
  - combined traditional work file 297
  - Conversion Mode 137
  - increase 139
  - lift 124
  - new-function mode (NFM) 137, 145, 261
  - preformats part 188
  - program directory 172
  - system 152
- DB2 BACKUP
  - System 165, 167
  - SYSTEM FULL job 166
  - SYSTEM utility job 167
- DB2 catalog 126, 136, 141–142, 208, 212, 217, 230–231, 238, 380, 383
- DB2 catalog and directory data sets 252
- DB2 data 2, 166, 193, 237, 255, 317, 344, 378
  - considerations 237
  - DFSMSshm 312
  - integrity 147
  - relevant code 315
- DB2 data set 2, 13, 165, 200, 202, 219, 230, 238, 384
- DB2 directory 144
- DB2 environment 3, 115, 213, 219, 238
  - Data Class 239
- DB2 I/O operations 339
- DB2 installation
  - panel 136
  - panel DSNTIPN 320
- DB2 STOGROUP 118, 122, 126, 207, 210–211, 221, 232, 257
  - non-DB2 volume 261
  - SMS class 222
  - specific entries 211
- DB2 storage
  - group 140, 310
  - objects 115
- DB2 system 136, 204, 310
  - checkpoint 351
  - copy pools 166
  - data set 313
  - recovery information 136
  - table 136
  - table space 145
  - task 321
- DB2 system table spaces 136
- DB2 table 118, 217, 233, 379
  - space 295
  - space set 118
- DB2 user
  - data 378
  - pool 217
  - table space 122, 231, 239, 294
- DB2 utility, REORG 314
- DB2 V8 42, 116, 340, 378
  - COPY TABLESPACE 153
  - PTF UK36548 179
- DB2 version

- 8 137, 344
- 9 117
- 9.1 127, 317, 339
- DDF 320
- DDM 39
  - failure 39
  - several blocks 51
- declared global temporary tables (DGTT) 139
- default value 136, 178, 240, 352
- deferred write threshold 349, 351
- DEFINE CLUSTER 128, 249
- device adapter
  - See DA
- device type 8, 15, 134, 196, 245, 336
- DFDSS 194, 201
- DFHSM 194, 201, 212, 214
- DFP 194–195, 220
- DFSMS xix, 3, 20, 22, 34, 127, 150–151, 193–194, 203, 218–219
- DFSMS/MVS 194, 217
- DFSMSsdfp 164, 194–195, 217, 227, 277
- DFSMSDSS 158, 305
- DFSMSdss 190, 194, 196–197, 213, 384
- DFSMSshm 160–161, 190, 194, 198, 213, 380, 384
- DFSMSopt 201–202
- DFSMSrmm 194, 201
- directory 111, 136, 209, 217, 219, 244, 382
- DISC time 329, 376–377
- disk architecture 11, 15
- disk controller 176, 204, 385
  - boot strap data sets 204
  - target response times 210
- disk drive 12
  - capacity 38
  - FATA 38
- disk space 14, 189, 311
  - requirement 173
  - significant savings 190
- disk subsystem 24, 26
- disk virtualization 47
- disk write burst 381
- DMTH 348
- DS Storage Manager 50
- DS Storage Manager *see* DS SM
- DS8000
  - AIX 46
  - architecture 6–7, 23, 44
  - DDM 48, 52
  - disk subsystem 45
  - FICON 148
  - FICON port concurrency 329
  - host adapter 149
  - multiple allegiance 29
  - PAV 29
  - performance 29, 149, 339
  - RAS 40
  - service 40
  - spares 40, 49
  - switched FC-AL 47
- DS8000 series 25, 44

- DSCB 22
- DSN1COPY 164
- DSNDB01 136, 144, 217, 233
- DSNDB01.SYSLGRNX 144
- DSNDB04.NONS MS 299
- DSNDB06 136, 142, 144, 217, 233
- DSNDB07 175, 178, 230, 233, 238, 380
- DSNTIPD 136
- DSNTIPN 144, 320
- DSNUVBBD - BACKUP (DB) 167
- DSS 196–197, 200, 210
  - components 196, 200
  - filtering 197
  - RELEASE
    - job 192
- DSSIZE 119–120, 209
- DSVCI 118
- dual copy 36, 260
- dual parity 37
- DUMPCCLASS 163
- DUMPCOND FR 167
- DWQT 349–351, 355
- dynamic PAV 27, 29
- dynamic prefetch 323, 340–341, 344–345
- Dynamic Volume Count (DVC) 208, 221, 238
- dynamic volume expansion 13, 34

## E

- EAS 21
- Easy Tier 17
- EAV xix, 4, 13, 29, 132–133, 209, 217, 232, 273
  - benefit 19
  - performance 29
- EAV volume 13, 18, 20–21, 295
  - cylinder-managed space 20
  - RFM format 130
- EMPTY cylinder 303
- ENDING AT 167
- Enterprise Storage Server
  - See ESS
- ERRORRANGE option 157
- ESCON 43, 238
- ESS 12, 16, 201
  - 800 41
  - technology 35
- EXCPEXIT 223, 250
- EXCPS 250
- expiration date 134, 208, 246
- EXT 175, 194, 224, 241, 303–304
- extended address volume
  - See EAV
- extended addressability (EA) 194, 241
- extended addressing space 20–21
- extended format (EF) 4, 132–133, 197, 200, 343, 381
- extended remote copy (XRC) 6, 14, 80
- extent
  - consolidation 187
  - definition 113
- Extent Constraint Removal
  - No 243

- option 179
- Yes 243
- extent pool 12, 51–52, 330, 335
  - DASD space 12
  - disk RPM 51
  - logical volumes 52
  - Multiple ranks 335
  - next rank 55
  - non-striped volumes 56
  - same characteristics 52
  - same RAID types 52
  - weighted average performance 335
- extent rotation 55
- extent size 50, 176–177
- extent space efficient (ESE) 56
- Extent Space Efficient Volumes 57
- extent type 50, 52

## F

- failed DDM 40
- Fast Log Apply (FLA) 168
- FC-AL switched 45
- FICON 31, 43, 68, 149, 238, 319, 343
- FICON channel 343, 376
  - link 328
  - saturation 377
  - total utilization 328
  - utilization 328
- FICON Express2 328
- FICON host
  - adapter 376
- FICON Open Exchange 329
- FICON port 376
- File and Storage Technologies (FAST) 12
- FILTLIST DB9ANOT1 258
- fixed block (FB) 50
- FlashCopy xix, 6, 18, 145, 200, 231, 238, 339, 381, 386
  - Consistency Group 89
  - data set 171
  - establish on existing RMC primary 89
  - incremental 161
  - options 171, 258
  - relationship 386
- FlashCopy SE 18, 33–34
- FlashCopy V2 162
- free space 22, 119, 199, 232, 379, 381
  - total amount 22
- FREESPACE 242
- FREESPC 250
- FUNCTION RC 167

## G

- GB extent size 51
- GDPS 22
- getpage 146, 321, 341
  - request 323, 341, 344
- gigabyte, or binary (GB) 16, 118, 243
- Global Copy 6, 14
- Global Mirror 6, 33

Guaranteed Space 126, 191, 197, 200, 211, 215, 232,  
238, 255, 261, 381  
    special Storage Class 222  
    Storage Class 205

## H

hard disk drive (HDD) 12  
head disk assembly (HDA) 23, 35  
help xix, xxii, 8, 28, 165, 213, 219, 339, 381  
HI-A-RBA 223, 250  
Hierarchical Storage Manager  
    See HSM  
High Performance FICON for System z 69  
high-level language (HLL) 43  
high-level qualifier (HLQ) 128, 173–174, 217, 233  
hit ratio 321, 340, 377  
HI-U-RBA 123, 223, 250, 383  
host adapter *see* HA  
hot spots 204, 235, 380  
HRECALL 200, 298  
HSM 194, 198, 213  
HyperPAV technology 29  
HyperSwap 22, 41  
Hypervisor 46

## I

I/O 11, 202, 317, 339  
    synchronous or asynchronous 321  
I/O activity 29, 321, 327  
    report 324–325  
    summary report 325  
I/O definition file (IODF) 27  
I/O operation 23, 321, 340, 343  
    detailed information 322  
    multiple pages 343, 349  
I/O performance 2, 125, 339  
    reporting 317  
I/O priority 13, 30, 319  
I/O priority queuing 32  
I/O Processor (IOP) 376  
I/O rate 19, 326  
I/O request 13, 210, 318, 341  
I/O supervisor (IOS) 12  
I/O supervisor queue 376  
IBM Redbooks publications xix, xxii  
IBM TotalStorage 7  
IBM Virtualization Engine TS7700 104  
IBM Virtualization Engine TS7740 105  
ICF catalogue 140, 160, 230, 312, 384  
    image copy entries 231  
IDC0001I Function 249  
IDC0002I IDCAMS 249  
IDC0181I DATACLASS 249  
IDC0181I MANAGEMENTCLASS 249  
IDC0181I STORAGECLASS 249  
IDC0508I DATA ALLOCATION Status 249, 251  
IDC0512I Name 249  
IDCAMS 118, 128, 136, 147, 208, 210, 219, 249, 298,  
305, 379

ALLOCATE 208  
DEFINE 208, 212, 252  
DEFINE CLUSTER 128  
LISTCAT 189  
IDRC 265  
IEFBR14 example 245  
image copy xix, 2, 9, 132–133, 142–144, 152, 163, 209,  
213, 217, 226–227, 230, 233, 237, 239, 252, 270, 340,  
342, 345, 380–381  
    options 153  
image copy data set 252, 380  
image copy data sets  
    naming convention 175  
immediate write threshold 348  
impact xix, 2, 5, 188, 340  
Improved Data Recording Capability 265  
Incremental FlashCopy 14, 161  
INDEX 124  
index xxii, 20, 22, 112–113, 116, 124, 198, 214, 216,  
231, 277, 298–299, 339–340, 343, 385  
index scan 343, 385  
index space 118, 125, 174, 232  
    creation 127, 174  
    image copies 152  
INDEXSPACE 125  
inhibit cache load 5  
INSTALLATION Exit 167  
instrumentation facility component (IFC) 317  
instrumentation facility interface (IFI) 317  
Intelligent Write Caching 67  
Interactive Storage Management Facility  
    *See* ISMF  
interval migration 199, 270  
IOP/SAP 326–327, 376  
IOS 23  
IOSQ time 376  
iSeries 43  
ISMF 129, 162, 195, 203, 218, 239, 299, 378  
    define or alter 239  
ISMF List 195  
ISMF panel 212, 221, 266  
    other default values 294  
    select option 5 222  
ISMF Profile 195  
    select option 0 218  
ISP 245  
IWC 6  
IWITH 350–352  
IXQTY DSNZPARMs 180

## K

KB 122, 340  
KB block 122  
KB page 119  
    size 122  
    size allocation 122  
KSDS file 246

## L

- large data set support 152
- large format data sets 4
- large object (LOB) 117, 347
- LCU 29, 31–32, 211, 217, 326
- LDS 118, 204
- LEAST 380
- LIKE 382
- linear data
  - set 120
- linear data set
  - See LDS
- Linux 44
- LIST 133, 230, 279, 354, 382
- list prefetch 321, 340–341, 345–346
- LISTC ENT 268
- LISTCAT output 123, 130, 209, 250, 383
- LISTVOL output 279
- LISTVTOC Vol 303
- LLQ EQ 234
- LOAD 143, 353, 382
- log xix, 2, 37, 133, 197, 225, 340, 380
- log read 152, 362
  - performance 362
- log record 143, 151, 323, 358
  - possible source 323
  - synchronous write 361
- logical control unit
  - See LCU
- logical partition
  - See LPAR
- logical subsystem 60
- logical subsystem *see* LSS
- logical volume 13, 15, 36, 45, 204
  - building blocks 50
  - conventional unit address 27
- logical volumes 25, 36, 188, 204
- LOGLOAD 350–351
- LOGONLY option 164
- low-level qualifier 217
- LPAR 24, 27, 45, 187, 235, 270, 277, 376
  - Hypervisor 46
- LRSN 168
- LSS 29–30, 60, 211, 217, 386

## M

- Management Class 126, 193, 195, 245
  - archive log data sets 197
  - option 3 230
- maximum number 54, 117, 345–346
- MAXLRECL 223, 250
- Media Manager 70
- METHOD 354, 386
- Metro Mirror 6, 14, 33–34
- Metro/Global Mirror 6, 14
- MGEXTSZ 176–177
- MGEXTSZ Yes 181
- MIDAW xix, 70, 148
  - facility 13

- mirroring 6, 36–37, 213
- ML2 198, 213, 229, 270
- Modified Indirect Data Address Word
  - See MIDAW
- MODIFY 144, 205
- most recently used (MRU) 153
- multicylinder unit (MCU) 21
- multiple allegiance 13, 23, 30, 319
- multitiered Storage Group 205, 212, 239
  - Group ACS routines 288
  - Storage Class 285

## N

- N/C 323
- name type 134, 225, 241
- naming convention 173, 175, 218, 380
  - table spaces and index spaces 173
- new-function mode (NFM) 138
  - DB2 V8 148
- NOMIGRATE 213
- non-production environment 213, 295
- nonvolatile storage 45
- non-VSAM data sets 116
- Nucleus Initialization Program (NIP) 28
- NULL Value 234, 301
- NVS 45, 381

## O

- OA06476 325
- OA10984 70
- object-level recovery 164
- OMEGAMON PE 202, 317, 342–343
- OMEGAMON XE 202
- OMEGAMON XE for Storage 194
- online transaction processing (OLTP) 146
- operating system (OS) 18, 47
- OUNT 254
- OW39951 382

## P

- page size 117, 124, 212, 341, 347
- parallel access volume
  - See PAV
- parallelism 24, 381
- PARTITION 234, 385
- partition-by-growth (PBG) 119
- partition-by-range (PBR) 119
- partitioned data set
  - See PDS
- partitioned data set (PDS) 116, 241
- partitioned data set extended (PDSE) 116
- path 15, 24, 45, 206, 210, 343–344
- PAV xix, 3, 13, 29, 31, 212, 214, 216, 230, 238, 297, 319, 327, 380
- PDS 142, 172, 196, 203, 237
- PDSE 116
- peer-to-peer remote copy
  - See PPRC



peer-to-peer remote copy extended distance

See PPRC-XD

PEND time 376

performance

z/OS 23, 139, 146, 339

physical control unit (PCU) 235, 289

physical copy 160

physical volume 204

PK05644 121

PK50113 179

PK70060 297

PK83108 211

PK93879 263

plan 141, 384

PM02528 139, 297

policy 152, 194, 203–204

power 7, 22, 55

PPRC 6, 14, 22, 79

PPRC-XD 14, 79

PQ88665 183

prefetch operation 344–345

prefetch quantity 341, 346–347

buffer pool 347

primary allocation 178, 216, 247

priority queuing 32

PRIQTY 120–121, 221, 265, 303, 383

PROC STORCLAS 258

production environment 148, 215, 270, 379

dual archive log data sets 150

## Q

QUALIFIER 384

## R

RAID 6, 12, 23, 34–35, 149, 204, 217, 226, 238

implementation 35, 39

RAID 10

AAL 40

drive failure 40

implementation 40

RAID 5 36, 40

implementation 39

RAID 6 14, 37

performance 38

RAMAC Array 35

RAMAC Virtual Array

See RVA

RANDOM 354

random write 38

ranks 50–51

read operations 342

rebuild time 38–40

recall 4, 200, 229, 231, 380

Record length 134, 227, 245

RECOVER 143, 159, 347, 353

recovery xix, 5, 136, 200, 229, 342, 380

recovery data sets 116, 147, 215

recovery strategy 148–149, 231

Redbooks website 395

Contact us xxii

redundant array of independent disks

See RAID

Referenced date 134, 227, 246

remote copy xix, 6

Remote Mirror 14

remote site 6, 153

REORG 143, 231, 298, 347, 353, 383, 385

reorg 56

REPORT 378–379, 381, 384–385

report 29, 216, 341, 348

Resource Measurement Facility

See RMF

response time 188, 210, 326, 340, 351

other component 376

RESTORE 160, 197, 298, 378, 380, 384

RETURN Code 167

return code 153, 227, 257

RIDs 345, 385

RMF 29, 317, 325, 381

RMF Magic 319

RMF report 29

RMM 201

role 148, 180

rotate volumes 56

RRSAF 320

RTS 384

RVA 37, 201, 209, 343, 386

## S

same ESS 162, 386

same time 40, 113, 202, 344

multiple failures 52

multiple PDSE members 113

same volume 3, 176, 204, 268, 292

sample naming structure for image copies 175

SAN 6, 43

SATA 38

scalability 22–23, 125, 145

SCDS 204, 234, 240, 299, 301

SCDS Name 240

Seascape architecture 6, 35

secondary allocation 179, 181, 216, 244

secondary cylinder 134, 211, 267

SECQTY 120, 124, 221, 265, 303, 383

SECQTY 50 129

SECQTY value 178

SELECT 272, 344, 384

Separation Profile 202, 238

SEQ 303–304, 354

sequential caching 5

sequential data 20, 116, 132, 202, 381

sequential prefetch 323, 340–341, 343

sequential prefetch threshold 348

server affinity 52

SET 167

SG Name 271

SHRLEVEL REFERENCE 133

copy 153

DSNUGULM 134

- IEC030I B37-04 133
- Scope 154
- sliding secondary allocation 177
- small computer system interface (SCSI) 36
- SMF 320
- SMF record type 42 202
- SMS 207
  - base configuration 204, 235, 289
  - classes 141, 150, 193, 195, 205, 207, 218, 220, 237, 300, 308
  - control data sets 204
  - data class 179, 207, 219
  - DB2 recovery data sets 237
  - examples of table space management 218
  - management class 142, 212, 229
  - management of DB2 databases 193
  - naming standard 217
  - storage class 23, 210, 222, 302
  - storage group 214–215
  - storage management policy 204–205
- SMS class 153, 205, 257, 269
  - LISTCAT output 308
- SMS configuration 203–205
- SMS construct 218, 237, 239
  - sample implementation 239
- SMS MANAGEMENT 126, 196, 237
  - OADR860I 305
  - OADR877I 305
- SMS management 198, 305
- SMS STORAGE Group
  - DB2 STOGROUPS 314
- SMS storage group 5, 118, 126, 199, 232, 271
- SMS volume
  - infrastructure 198
- SnapShot 157, 201, 386
- solid-state drive
  - See SSD 12
- sort 164, 233, 345, 380
- source control data set (SCDS) 204, 299
- source volume 161, 386
  - exact size 162
- space constraint relief (SCR) 209, 211, 241
  - Data Class specification 261
- Space Efficient (SE) 18, 56
- space management 2–3, 194, 196, 198–199, 215
- spares 39–40
- sparing 39
- specified table space, clone table 170
- SPLITS-CA 250
- SPLITS-CI 250
- SPTH 348–349
- SQL 117, 220, 340, 384–385
- SSD 12, 15
- START WITH 380
- STATEMENT 133, 385
- STOGROUP 125
- storage administrator 2, 11, 161, 195, 239, 339, 378
- storage capacity 31, 44
- Storage Class 195, 218, 239
  - extended format VSAM data sets 209

- Guaranteed Space 206
- ISMF option 5 258
  - specified criteria 215
- Storage Group ACS routine searches 225
- Storage Class ACS
  - routine 258, 301
  - routine panel 233
  - routine work 233
- storage complex 46
- storage device 34
- Storage Facility Image (SFI) 13
- Storage Group 5–6, 125–126, 205, 214–215, 221, 379
  - appropriate volume 214
  - HIGH and LOW values 231
  - HIGH threshold 284
  - initial panel 274
  - LOW values 273
  - new type 162
  - option 6 273
  - SMS-managed attributes 176
  - specific volume 211
  - types 214
- Storage Group (SG) 2, 193, 195–196, 239, 337, 378
- storage LPAR 46
- storage management 2–3, 194, 310, 317
  - issue 385
  - Report 196
  - Subsystem 140
- Storage Management Facility 195, 203
- Storage Management Subsystem (SMS) 162
- Storage Pool Striping (SPS) 6, 12, 52, 261, 334–335
- storage server 5, 7, 35, 43
- storage unit 46
- stored procedures 382
- stripe 37, 39, 183, 226, 260
  - size 39, 264
- striping xix, 4, 36–37, 148, 188, 210, 212, 219, 222, 238
- SUSIBM.SYSCOPY 142
- suspend 165
- Sustained Data Rate (SDR) 212, 259–260
- SVOLARC 211
- switched FC-AL 48
- Synchronous I/O 340
- synchronous read 342, 344, 354
- synchronous write 351
- Sys Group (SG) 271, 276
- Sys/Sys Group Name 271
- SYSIBM.SYSCOPY 136
  - obsolete information 160
- SYSIBM.SYSLGRNX 136, 144
- SYSIN DD 155
- SYSLGNRX entry 144
- SYSPRINT DD SYSOUT 168, 245
- system able spaces 116
- System Data Mover (SDM) 101
- system managed storage (SMS) 193–194, 237
- System Management Facilities (SMF) 329
- System Resources Manager (SRM) 214
- System Services Runtime environment (SSRE) 42
- System z servers

- CONN time 326, 376–377
- DISC time 329, 376–377
- IOSQ time 376
- PEND time 376
- system-level backup 161, 163

## T

- table 4, 116–117, 128, 193, 209, 216, 218, 223, 239, 303, 340, 342, 344, 379
- table space 116–119, 129, 216, 221–224, 277, 281, 284, 304, 325, 341, 343–344, 346, 381
  - complete backups 153
  - creation 127, 284
  - following types 118
  - image copies 153
  - same page 118
  - single and multiple tables 118
  - space usage 120
  - Storage Group 277
  - system 116
  - user 116, 217, 230, 238
- TABSPACE DB1.TS1 156
- tape drive 8, 155, 188, 201
- tape virtualization 8
- tape volume cache (TVC) 15
- target volume 161–162, 386
  - same location 386
- TASKID 001 158, 167, 305
- TASKID 002 167
- TB 20, 119
- test cases 29
- Thin Provisioning 54
- TIME 320
- Time of Day (TOD) field 161
- timestamp 126
- TOKEN 167
- token 161
- topology 5
- total cost of ownership (TCO) 9
- track 5, 15, 85, 121, 196, 232, 235, 378, 381
- track size 122
- Track Space Efficient 57
- TRACK-MANAGED Space 303
  - EMPTY TRACKS 303
- track-managed space 20, 215, 303
- TS1130 105
- TS3400 105
- TS3500 105
- TS7700 Virtualization Engine 104

## U

- UCAT.DB9A 123, 223, 243
- UCAT.DB9A STORCLAS 258
- UCB 26, 29, 293, 379
- UK56045 139
- UK56046 139
- UNALLOCATED VIRS 303
- UNICODE 320
- UNIQUE 124, 223, 243, 380

- unit control block (UCB) 26, 311, 327
- unused space 146, 197, 202, 239, 270
- UPDATE 384
- UQ89517 183
- USER 192, 213, 225, 305, 380
- user 195
  - groups 5, 116, 200
- user table space 116
- user-controlled threshold 348
- user-defined table space 128

## V

- VALIDATE 205, 382
- VCAT DB9AU 223, 303
  - DATACLAS DB2EFEA 126
- VDWQT 349–351, 355, 382
- vertical deferred write threshold 351
- VIEW 384
- Virtual Tape Server (VTS) 8
- virtual volume 15
- virtualization
  - arrays 49
  - concepts 46
  - definition 46
  - extent pools 51
  - hierarchy 50
  - ranks 50
- VOLATILE 381
- VOLSER 123–124, 209, 223–224, 252, 254, 298, 379
- Volume Count 221
  - Data Class entry 221
- Volume Count (VC) 208, 221, 238
- volume group 62
- volume JI8118 302, 304
  - FOLLOWING DATA SETS 305
- volume JI8119 303
  - FOLLOWING DATA SETS 306
- volume manager 57
- Volume Mount Analyzer (VMA) 316
- VOLUME SBOX1Y 249, 298
  - IDC0508I DATA ALLOCATION STATUS 292
  - IDC0509I INDEX ALLOCATION STATUS 249
- volume selection 4, 126, 195, 276
  - SMS control 215
- volumes
  - CKD 53
  - logical 53
- VSAM data 2, 13, 19, 118, 140
- VSAM data set 13, 19
- VSAM striping 13, 226–227, 261
- VTOC 20, 22, 34, 195, 197–198, 214, 272, 298–299, 379
- VTOC index 302
- VVDS 198, 298

## W

- WFDBSEP 139
- WLM 13, 30, 339
- work file 137, 212, 297, 351
- work file database 138–139

- WORKFILE 138–139
  - database, space use 139
- WORKFILE database
  - space utilization 139
  - table spaces 137
- workload 3, 5, 27–28, 149
- Workload Manager 27, 339
- write frequency 353, 355
- write operations 165, 349
- write penalty 36
- write quantity 346, 353
- write threshold 348

## **X**

- XML table
  - space 120
  - spaces share 120
- XRC 6

## **Z**

- z/OS 1.7 152, 381
- z/OS environment 319
  - application I/O 319
- z/OS Global Mirror 6, 33–34
- z/OS Metro/Global Mirror 6
- z/OS V1R10 4, 16
  - Large volumes 16
- z/OS Workload Manager 27
- zEnterprise System 69
- zHPF 69
- zIIP 101
- zSeries servers, PAV 327



## DB2 9 for z/OS and Storage Management

(0.5" spine)  
0.475" <-> 0.873"  
250 <-> 459 pages







**Redbooks®**

# DB2 9 for z/OS and Storage Management

**Manage all DB2 objects with system-managed storage**

**Make productive use of storage servers architecture**

**Use tapes and virtual tapes for best price performance**

This IBM Redbooks publication can help you tailor and configure DFSMS constructs to be used in an IBM DB2 9 for z/OS environment. In addition, it provides a broad understanding of new disk architectures and their impact in DB2 data set management for large installations.

This book addresses both the DB2 administrator and the storage administrator. The DB2 administrator can find information about how to use DFSMS for managing DB2 data sets; the storage administrator can find information about the characteristics of DB2 data sets and how DB2 uses the disks. This book describes optimal use of disk storage functions in DB2 for z/OS environments that can best make productive use of the synergy with I/O subsystem on IBM System z.

This book covers the following topics:

- ▶ Using SMS to manage DB2 catalog, log, data, indexes, image copies, archives, work files
- ▶ Taking advantage of IBM FlashCopy for DB2 utilities, striping, copy pools
- ▶ Setting page sizes and using sliding allocation
- ▶ A description of PAV, MA, MIDAW, EF, EA, EAV, zHPF and why they are helpful
- ▶ Compressing data and the use disk and tape for large data sets
- ▶ Backup and restore, and remote copy services

## **INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION**

### **BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
**[ibm.com/redbooks](http://ibm.com/redbooks)**